AD 746484

July 1972
Technical Report 7201

TRANSVERSE GRAVITY EFFECTS ON A FULLY CAVITATING HYDROFOIL RUNNING BELOW A FREE SURFACE

Bruce E. Larock
Davis, California

11

This research was carried out under the
Naval Ship Systems Command
General Hydromechanics Research Program
Subproject SR 009 01 01, administered by the
Naval Ship Research and Development Center,
Office of Naval Research Contract N00014-72-C-0109

TRANSVERSE GRAVITY EFFECTS ON A FULLY CAVITATING HYDROFOIL RUNNING BELOW A FREE SURFACE

Bruce E. Larock
Davis, California

Technical Report 7201
July 1972

This research was carried out under the
Naval Ship Systems Command
General Hydromechanics Research Program
Subproject SR 009 01 01, administered by the
Naval Ship Research and Development Center,
Office of Naval Research Contract N00014-72-C-0109

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY (Corporate author) Dr. Bruce E. Larock 1412 Drake Drive, Apt. A Davis, California 95616	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3 REPORT TITLE

TRANSVERSE GRAVITY EFFECTS ON A FULLY CAVITATING HYDROFOIL RUNNING BELOW A FREE SURFACE

4 DESCRIPTIVE NOTES (Type of report and inclusive dates)

Final Technical Report, December 1971-July 1972

5 AUTHOR(S) (First name, middle initial, last name)

Bruce E. Larock

6 REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
July 1972	64	14

8a. CONTRACT OR GRANT NO N00014-72-C-0109 b. PROJECT NO. SR 009 01 01 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) TR 7201 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10 DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11 SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Ship Research and Development Center, Dept. of the Navy, Washington, D.C. 20034

13 ABSTRACT

Equations are presented which describe the fully cavitating flow of fluid past a flat plate hydrofoil running below a free surface. Transverse gravity field effects are included in the analysis. The equations are developed by the use of complex function theory and Tulin's double-spiral-vortex cavity model. Two FORTRAN IV computer programs have been developed to evaluate the equations. Features and use of these programs are discussed, and program listings are presented in the appendix. (U)

Preceding page blank

DD FORM 1 NOV 65 1473 REPLACES DD FORM 1473, 1 JAN 64, WHICH IS OBSOLETE FOR ARMY USE

UNCLASSIFIED

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Hydrofoil						
Fully cavitating flow						
Lift						
Drag						
Gravity						
Free Surface						
Iteration						

ABSTRACT

Equations are presented which describe the fully cavitating flow of fluid past a flat plate hydrofoil running below a free surface. Transverse gravity field effects are included in the analysis. The equations are developed by the use of complex function theory and Tulin's double-spiral-vortex cavity model. Two FORTRAN IV computer programs have been developed to evaluate the equations. Features and use of these programs are discussed, and program listings are presented in the appendix.

CONTENTS

ILLUSTRATIONS

LIST OF SYMBOLS

A_{11}	coefficient, see Eqs. 22, 24
A_{12}	coefficient, see Eqs. 23, 25
B_1	coefficient independent of gravity, see Eqs. 22, 26
B_{1G}	coefficient dependent on gravity, see Eqs. 22, 28
B_2	coefficient independent of gravity, see Eqs. 23, 27
B_{2G}	coefficient dependent on gravity, see Eqs. 23, 29
C_D	drag coefficient, see Eq. 52
C_L	lift coefficient, see Eq. 52
d	submergence depth
D	drag
e	2.71828... = base of natural logarithms
F	Froude number, see Eq. 4
g	acceleration of gravity
$G_{1c}(t)$	gravity effect of cavity on plate, see Eqs. 30, 32
$G_{1s}(t)$	gravity effect of free surface on plate, see Eqs. 30, 33
$G_{2c}(t)$	gravity effect of cavity on ω_I, see Eqs. 38, 40
$G_{2s}(t)$	gravity effect of free surface on ω_I, see Eqs. 38, 41
$G_{3c}(t)$	gravity effect of cavity on γ, see Eqs. 44, 45
$G_{3s}(t)$	gravity effect of free surface on γ, see Eqs. 44, 46
H(t)	solution to homogeneous Hilbert problem
i	imaginary unit = $(-1)^{1/2}$
I_c	integral related to C_D, C_L; see Eq. 56
Im( )	imaginary part of complex quantity
I_p	integral related to plate length, see Eq. 48
ℓ	plate length
L	lift
m	integer, defined following Eq. 60

p	pressure
p_a	atmospheric pressure
p_c	cavity pressure
p_o	reference pressure
q	fluid speed
q_c	local fluid speed on cavity boundary
q_o	magnitude of undisturbed reference velocity at infinity
q_s	local fluid speed on free surface
$Q(t)$	solution to Hilbert problem
$Re(\ )$	real part of complex quantity
$S_1(t)$	function, see Eqs. 30, 31
$S_2(t)$	function, see Eqs. 38, 39
t	upper half-plane variable
t_o	initial t-value in an integral expression
t_u	large, real, positive, finite t-value; see Eq. 15 ff.
t_A	point A in t-plane
t_D	point D in t-plane
t_E	point E in t-plane
W	$\phi + i\psi$ = complex potential
x	horizontal coordinate in physical plane
x_o	initial x-value in an integral expression
x_B, y_B	coordinates of plate stagnation point
x_C, y_C	coordinates of trailing edge of plate
y	vertical coordinate in physical plane
y_c	y-coordinate of cavity boundary
y_o	initial y-value in an integral expression
y_s	y-coordinate of free surface
z	$x + iy$ = physical plane

z_s	coordinate on free surface
z_w	coordinate on upper wake
α	attack angle
β	location of point F in t-plane
$\gamma(t)$	imaginary part of ω on wakes and free surface, see Eqs. 42-44
δ	radius of arc around $t = \beta$ in t-plane
ε	small, positive number ($\varepsilon << 1$)
η	dummy variable
θ	argument of velocity, Eq. 8; also arc coordinate, see p. 23
π	3.14159...
ρ	constant fluid density
σ	cavitation number
ϕ	velocity potential
ϕ_D	velocity potential at point D
ϕ_E	velocity potential at point E
χ	stream function
ψ_o	stream function value on free surface
ω	logarithm of normalized complex velocity
ω_I	imaginary part of ω on cavity boundary, see Eqs. 36, 36
ω_R	real part of ω on cavity boundary, see Eqs. 36, 37

INTRODUCTION

Cavitating flow theory has advanced rapidly in numerous directions within the last 15 years in response to varied stimuli. Most important is the increasing use, and interest in further enlarging the capabilities, of hydrofoil craft of varying designs. Organizations have been willing to sponsor research in the area, which has resulted in new approaches to the solutions of problems, many of which depend heavily on the digital computer to make possible the numerical evaluation of results. These trends in research are evident in the current project, which builds upon several earlier projects.

Three lines of previous research should be noted. Much theoretical work has sought to develop linear and nonlinear theories to describe accurately the behavior of flat-plate hydrofoils in the absence of gravity effects, e.g., see Refs. (1-8). In 1969 the experiments of Dinh (9) indicated that Larock and Street (7) at that time provided the most accurate theory for a flat-plate foil in a fluid of infinite extent. Some of the above works treated a foil which operates near a free surface. Larock and Street (8) presented a nonlinear theory for this problem which indicated, among other results, that (a) linearized theory somewhat overpredicted the lift coefficient, and (b) as a consequence of the lift on the foil and (it is believed) the neglect of gravity, the vertical coordinate of the free surface became unbounded far upstream.

A small number of investigators, realizing that in some circumstances the effects of gravity are not insignificant, have conducted studies of various gravity-affected cavity flows

(10-14). Of these investigations, only that of Larock and Street (14) does not depend on a special symmetry or approximate boundary conditions to effect a solution; it presents a nonlinear theory for a fully cavitating flat-plate hydrofoil acted upon by a transverse gravity field in a fluid of infinite extent. To date, however, the theoretical behavior of a hydrofoil operating in the fully cavitating flow state beneath a free surface, with a consideration of the effect of a transverse gravity field, is still unknown.

The present research addresses itself to this last problem by combining ideas developed previously in two papers by Larock and Street, Refs. (7) and (14).

SCOPE AND PURPOSE OF RESEARCH

This work extends the 1967 free-surface nonlinear theory of Larock and Street (8) for flow past a fully cavitating flat-plate hydrofoil by incorporating the effects of a transverse gravity field into the theory. The development proceeds along much the same line as that of Larock and Street's earlier gravity theory (14). The Riemann-Hilbert technique is used in conjunction with Tulin's double-spiral-vortex cavity model to seek the solution. The solution for the gravity-affected problem is found by iteration; the nongravity solution (8) is used in the initial computation of the additional gravity terms in the final solution.

In this project the equations describing the problem are completely developed and presented in this report. Two FORTRAN IV computer programs have been written to evaluate these equations; a listing of each appears in an appendix to this report. Another section of this report is intended to serve as a user's manual for these programs. Finally, the Scientific Officer for this project will be supplied with a card deck for these programs.

The Naval Ship Research and Development Center is expected to use the results of this project for the purpose of determining the effects of gravity on the relations between lift, drag and submergence, and on cavity shape and free surface configuration.

THEORY

The underlying structure for this theory has been presented previously (7); the basic approach for incorporating the transverse gravity field into the theory has also been presented several years ago (14). The following exposition is intended to be self-contained but rather concise; the interested reader is referred to (7) and (14) for extended discussions.

The Tulin nonlinear, double-spiral-vortex model (6,8) is employed here because its features are useful in the subsequent conformal- mapping procedure.

Figure 1 shows the physical plane (z-plane) for the flow past a flat plate running beneath the free surface and trailing a cavity modeled by the Tulin model. The plate ABC is inclined at an angle α relative to the parallel fluid flow of undisturbed speed q_o which surrounds it. The nose of the plate is chosen to be the coordinate origin with the x-axis drawn parallel to the undisturbed flow. Point A is submerged a distance d below the undisturbed free surface, denoted by I_2. The flow separates smoothly from the ends of the plate. The free surfaces bounding the cavity, of cavity pressure p_c and local speed $q_c(y)$, end in a pair of double-spiral vortices at points D and E. These streamlines spiral into and terminate at these points. There the flow speed discontinuously jumps from the cavity speed q_c to the undisturbed speed q_o, and the wake boundary spirals back outward and proceeds to downstream infinity at F_1 and I_1 (this behavior is responsible for the name of the model). Tulin remarks that this velocity discontinuity accounts partially for the pressure-recovery loss caused by the turbulence that accom-

panies cavity collapse. The local speed along the free surface I_2F_2 is $q_s(y)$. It is convenient to select q_o as the characteristic velocity for the problem and later normalize it to unity.

For this problem the fluid is assumed to be inviscid, incompressible and homogeneous. The flow is steady, irrotational and two-dimensional with surface tension effects neglected. In this case the Bernoulli equation is

$$p + \frac{1}{2}\rho q^2 + \rho g y = \text{Constant} \tag{1}$$

Here p is the pressure, ρ is the fluid density, q is the fluid speed, g is the acceleration of gravity, and y is the vertical distance to some point in the flow from the coordinate origin. The Constant can be evaluated at a convenient reference point, which is currently chosen to be far upstream at y = 0; conditions at this point are denoted by a subscript o. Thus, if Eq. 1 is applied between the reference point and some point on the cavity (subscript c), one obtains

$$p_o + \frac{1}{2}\rho q_o^2 = p_c + \rho g y_c + \frac{1}{2}\rho q_c^2 \tag{2}$$

By introducing the cavitation number

$$\sigma = \frac{p_o - p_c}{\frac{1}{2}\rho q_o^2} \tag{3}$$

and the Froude number F, defined by

$$F^2 = \frac{q_o^2}{g\ell} \tag{4}$$

where ℓ is the plate length, one finds that the ratio of fluid speeds q_c/q_o can be written

$$\frac{q_c}{q_o} = \left[1+\sigma-\frac{2}{F^2}\,\frac{y_c}{\ell}\right]^{1/2} \tag{5}$$

Far upstream where the flow speed is uniformly q_o, one finds the relation between p_o and the atmospheric pressure p_a at the free surface to be

$$p_o = p_a + \rho g d \tag{6}$$

Then by applying the Bernoulli equation between the free surface and the reference point, the speed ratio q_s/q_o is found to be

$$\frac{q_s}{q_o} = \left[1+\frac{2}{F^2}\left(\frac{d-y_s}{\ell}\right)\right]^{1/2} \tag{7}$$

For the assumed flow and model, the plane of the complex potential $W = \phi + i\psi$ can be drawn, Fig. 2. The physical and complex potential planes are uniquely related by

$$\frac{1}{q_o}\,\frac{dW}{dz} = \frac{q}{q_o}e^{-i\theta} = e^{\omega} \tag{8}$$

where ω is the logarithm of the normalized complex velocity with argument θ. Alternatively, one has

$$\omega = \ell n(q/q_o) + i(-\theta). \tag{9}$$

Rearrangement of Eq. 8 then gives the physical-plane configuration as

$$z = \frac{1}{q_o}\int e^{-\omega(t)}\frac{dW}{dt}\,dt \tag{10}$$

when all variables are expressed in terms of the same variable, called t. The t-plane is an upper half-plane, with the boundary

of the flow domain lying on the real axis, Fig. 3. (Henceforth q_o is set to unity.)

To express W and ω as functions of t, first W is mapped to the half-plane by use of the Schwarz-Christoffel transformation, and then ω is directly constructed in the t-plane. The function W(t) is made unique by selecting the following three-point correspondence:

$$\begin{array}{llll} B: & W = 0 & t = 0 & \\ C: & W = \exp(2\pi i) & t = -1 & (11) \\ I: & W \to \infty & t \to \infty & \end{array}$$

If $t = \beta$ is the point at downstream infinity (point F) and boundary F_2I_2 is used to evaluate the constant of integration, one finds

$$W(t) = -\frac{\psi_o}{\pi\beta}\,[t+\beta\ \ell n(1-t/\beta)] \qquad (12)$$

The parameters ψ_o and β are related by considering point C; the result is

$$\psi_o = \pi\beta/[1-\beta\ \ell n(1+1/\beta)] \qquad (13)$$

Finally, if one requires that $\phi_E = \phi_D$, as was done in Ref. (8), another parameter relation is obtained as

$$t_E/\beta + \ell n(1-t_E/\beta) = t_D/\beta + \ell n(1-t_D/\beta) \qquad (14)$$

One uses the Riemann-Hilbert technique to construct $\omega(t)$. Initially one hypothesizes that the following boundary data are known along the entire real line in the t-plane:

$$
\begin{array}{lll}
\mathrm{Re}(\omega) = 0 & & -\infty < t < t_D \\
\mathrm{Re}(\omega) = \frac{1}{2}\ln\left[1+\sigma-\frac{2}{F^2}\frac{y_c(t)}{\ell}\right] & & t_D < t < -1 \\
\mathrm{Im}(\omega) = \alpha & & -1 < t < 0 \\
\mathrm{Im}(\omega) = \alpha - \pi & & 0 < t < t_A \\
\mathrm{Re}(\omega) = \frac{1}{2}\ln\left[1+\sigma-\frac{2}{F^2}\frac{y_c(t)}{\ell}\right] & & t_A < t < t_E \\
\mathrm{Re}(\omega) = 0 & & t_E < t < \beta \\
\mathrm{Re}(\omega) = \frac{1}{2}\ln\left[1+\frac{2}{F^2}\left(\frac{d-y_s(t)}{\ell}\right)\right] & & \beta < t < t_u \\
\mathrm{Re}(\omega) = 0 & & t_u < t < \infty
\end{array}
\tag{15}
$$

In the above it is assumed that $y_s(t)$ approaches d as t grows progressively larger, and the difference $(d-y_s(t))$ is negligibly small for $t > t_u$, t_u being some reasonably large, but finite, value of t.

The current technique requires a knowledge of the imaginary part of a function, say Q(t), on the entire real line. This is achieved by the definition

$$Q(t) = \omega(t)/H(t) \tag{16}$$

where H(t) is a solution to the homogeneous Hilbert problem. For the current problem one may select (8)

$$H(t) = -i[(1+t)(t-t_A)]^{1/2} \tag{17}$$

The solution for $\omega(t)$ is

$$Q(t) = \frac{\omega(t)}{H(t)} = \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{\mathrm{Im}[Q(\eta)]d\eta}{\eta-t} \tag{18}$$

Ref. (8) proves that no power series terms should be added to this expression.

In writing the full expression for $\omega(t)$, it is advantageous to rewrite one of the logarithmic terms in Eq. 15 in the following way:

$$\ell n\left[1+\sigma-\frac{2}{F^2}\frac{y_c(t)}{\ell}\right] = \ell n(1+\sigma)+\ell n\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(t)}{\ell}\right] \tag{19}$$

The integral form for $\omega(t)$ is therefore composed of seven terms:

$$\frac{\omega(t)}{H(t)} = -\frac{1}{2\pi}\ell n(1+\sigma)\int_{t_D}^{-1}\frac{d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}$$

$$+\frac{1}{2\pi}\ell n(1+\sigma)\int_{t_A}^{t_E}\frac{d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}$$

$$+\frac{\alpha}{\pi}\int_{-1}^{t_A}\frac{d\eta}{(\eta-t)[(1+\eta)(t_A-\eta)]^{1/2}}-\int_{0}^{t_A}\frac{d\eta}{(\eta-t)[(1+\eta)(t_A-\eta)]^{1/2}}$$

$$-\frac{1}{2\pi}\int_{t_D}^{-1}\frac{\ell n\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}$$

$$+\frac{1}{2\pi}\int_{t_A}^{t_E}\frac{\ell n\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}$$

$$+\frac{1}{2\pi}\int_{\beta}^{t_u}\frac{\ell n\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}. \tag{20}$$

The requirement that the flow be horizontal at infinity

yields two additional conditions that contribute to a unique specification of all flow parameters. The conditions (8) are

$$\begin{aligned} \mathrm{Im}[\omega(\beta)] &= 0 \\ \mathrm{Im}[\omega(\infty)] &= 0 \end{aligned} \tag{21}$$

When Eqs. 21 are applied to Eq. 20, the resulting equations can be displayed in the form

$$A_{11}\ln(1+\sigma) + \alpha = B_1 + B_{1G} \tag{22}$$

$$A_{21}\ln(1+\sigma) + \alpha = B_2 + B_{2G} \tag{23}$$

where

$$A_{11} = \frac{1}{2\pi}\left\{\ln\left[\frac{\left(\frac{\beta-t_A}{1+\beta}\right)^{1/2}+\left(\frac{t_E-t_A}{1+t_E}\right)^{1/2}}{\left(\frac{\beta-t_A}{1+\beta}\right)^{1/2}-\left(\frac{t_E-t_A}{1+t_E}\right)^{1/2}}\right] - \ln\left[\frac{\left(\frac{1+\beta}{\beta-t_A}\right)^{1/2}+\left(\frac{1+t_D}{t_D-t_A}\right)^{1/2}}{\left(\frac{1+\beta}{\beta-t_A}\right)^{1/2}-\left(\frac{1+t_D}{t_D-t_A}\right)^{1/2}}\right]\right\} \tag{24}$$

$$A_{21} = \frac{1}{\pi}\ln\left[\frac{(1+t_E)^{1/2}+(t_E-t_A)^{1/2}}{(t_A-t_D)^{1/2}+(-(1+t_D))^{1/2}}\right] \tag{25}$$

$$B_1 = \frac{\pi}{2} - \sin^{-1}\left[\frac{\beta(1-t_A)-2t_A}{\beta(1+t_A)}\right] \tag{26}$$

$$B_2 = \frac{\pi}{2} - \sin^{-1}\left[\frac{1-t_A}{1+t_A}\right] \tag{27}$$

$$B_{1G} = \frac{1}{2\pi}[(1+\beta)(\beta-t_A)]^{1/2}\left\{-\int_{t_D}^{-1}\frac{\ln\left[1-\frac{2}{F^2}\,\frac{1}{1+\sigma}\,\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-\beta)[(1+\eta)(\eta-t_A)]^{1/2}}\right.$$

$$+\int_{t_A}^{t_E} \frac{\ln\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-\beta)[(1+\eta)(\eta-t_A)]^{1/2}} + \int_{\beta}^{t_u} \frac{\ln\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]d\eta}{(\eta-\beta)[(1+\eta)(\eta-t_A)]^{1/2}}\Bigg\} \tag{28}$$

and

$$B_{2G} = \frac{1}{2\pi}\Bigg\{\int_{t_D}^{-1} \frac{\ln\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{[(1+\eta)(\eta-t_A)]^{1/2}}$$

$$-\int_{t_A}^{t_E} \frac{\ln\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{[(1+\eta)(\eta-t_A)]^{1/2}} - \int_{\beta}^{t_u} \frac{\ln\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]d\eta}{[(1+\eta)(\eta-t_A)]^{1/2}}\Bigg\} \tag{29}$$

The actual setting of flow parameters proceeded in the following order:

1. Either ψ_o or β is prescribed, and the other parameter is found from Eq. 13.
2. To take advantage of the linearity of Eqs. 22 and 23 in $\ln(1+\sigma)$ and α, these equations are solved repeatedly for $\ln(1+\sigma)$ and α for assigned arrays of values for t_A and t_D; t_E is found from Eq. 14 for each given t_D and β.
3. An appropriate preliminary selection of t-plane parameters, i.e. t_A, t_D, t_E, and β, can then be made from the tabular arrays of parameters which were generated in step 2.

The next section of this report presents all the equations which together constitute a complete solution to the problem.

The methodology for computing this solution then follows. It is there that the treatment of the gravity terms B_{1G} and B_{2G} which obviously cannot initially be computed with the other terms (since $y_c(t)$, $y_s(t)$, t_u, and F^2 are not yet known), will be discussed.

RESULTS

Results of the theory include the physical-plane configuration for the plate, cavity and wake boundaries and the free surface, and also the lift and drag coefficients.

On the foil, $-1<t<t_A$, and Eq. 20 can be evaluated to give $\omega(t)$ as

$$\omega(t) = i\alpha + S_1(t) + G_{1c}(t) + G_{1s}(t)$$

$$+ \ell n\left[\frac{(t_A-t)^{1/2} - [t_A(1+t)]^{1/2}}{(t_A-t)^{1/2} + [t_A(1+t)]^{1/2}}\right] \quad (30)$$

where

$$S_1(t) = \frac{1}{2\pi}\ell n(1+\sigma)\left\{\pi + \sin^{-1}\left[\frac{(1+2t_D-t_A)t+(1-t_A)t_D-2t_A}{(t-t_D)(1+t_A)}\right]\right.$$

$$\left. + \sin^{-1}\left[\frac{(1+2t_E-t_A)t+(1-t_A)t_E-2t_A}{(t_E-t)(1+t_A)}\right]\right\} \quad (31)$$

$$G_{1c}(t) = \frac{1}{2\pi}[(1+t)(t_A-t)]^{1/2}\left\{\int_{t_A}^{t_E} \frac{\ell n\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}\right.$$

$$\left. - \int_{t_D}^{-1} \frac{\ell n\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}\right\} \quad (32)$$

and

$$G_{1s}(t) = \frac{1}{2\pi}[(1+t)(t_A-t)]^{1/2}\int_{\beta}^{t_u} \frac{\ell n\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}}$$

(33)

It can be shown that

$$G_{1c}(-1) = \frac{1}{2}\ln\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(-1)}{\ell}\right] \tag{34}$$

and

$$G_{1c}(t_A) = \frac{1}{2}\ln\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(t_A)}{\ell}\right] = 0 \tag{35}$$

since $y_c(t_A) = 0$.

For the two free surfaces bounding the cavity,

$$\omega(t) = \omega_R(t) + i\omega_I(t) \tag{36}$$

with

$$\omega_R(t) = \frac{1}{2}\ln(1+\sigma) + \frac{1}{2}\ln\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(t)}{\ell}\right] \tag{37}$$

and

$$\omega_I(t) = \alpha - 2\left\{\frac{\pi}{2}-\tan^{-1}\left[\frac{t-t_A}{t_A(1+t)}\right]^{1/2}\right\} + S_2(t)+G_{2c}(t)+G_{2s}(t) \tag{38}$$

For the lower free surface $t_D \leq t \leq -1$, and for the upper free surface $t_A \leq t \leq t_E$. The last three terms in Eq. 38 are defined as

$$S_2(t) = \frac{1}{2\pi}\ln(1+\sigma)\left\{\ln\left[\frac{\left(\frac{t_E-t_A}{1+t_E}\right)^{1/2}+\left(\frac{t-t_A}{1+t}\right)^{1/2}}{\pm\left\{\left(\frac{t_E-t_A}{1+t_E}\right)^{1/2}-\left(\frac{t-t_A}{1+t}\right)^{1/2}\right\}}\right]\right.$$

$$\left. - \ln\left[\frac{\left(\frac{1+t}{t-t_A}\right)^{1/2}+\left(\frac{1+t_D}{t_D-t_A}\right)^{1/2}}{\pm\left\{\left(\frac{1+t}{t-t_A}\right)^{1/2}-\left(\frac{1+t_D}{t_D-t_A}\right)^{1/2}\right\}}\right]\right\} \tag{39}$$

$$G_{2c}(t) = -\frac{t}{2\pi}\left[\left(1+\frac{1}{t}\right)\left(1-\frac{t_A}{t}\right)\right]^{1/2}\left\{ \fint_{t_A}^{t_E} \frac{\ell n\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}} \right.$$

$$\left. -\fint_{t_D}^{-1} \frac{\ell n\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}} \right\} \qquad (40)$$

and

$$G_{2s}(t) = -\frac{t}{2\pi}\left[\left(1+\frac{1}{t}\right)\left(1-\frac{t_A}{t}\right)\right]^{1/2}\int_{\beta}^{t_u} \frac{\ell n\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]d\eta}{(\eta-t)[(1+\eta)(\eta-t_A)]^{1/2}} \qquad (41)$$

The slash on the integral signs in Eq. 40 signifies that the interval $(t-\varepsilon,\ t+\varepsilon)$ is to be deleted from the integration, where ε is a suitably small, positive number. In Eq. 39 use the (+) for the upper surface and the (-) for the lower surface where $(\pm)$ signs appear.

For the wake boundaries and the free surface one finds

$$\omega(t) = i\gamma(t) + \frac{1}{2}\ell n\left[1+\frac{2}{F^2}\left(\frac{d-y_s(t)}{\ell}\right)\right], \text{ for } t > \beta \qquad (42)$$

and

$$\omega(t) = i\gamma(t), \text{ for } t < \beta \qquad (43)$$

where

$$\gamma(t) = \frac{1}{2\pi}\ell n(1+\sigma)\left\{\ell n\left[\frac{\left(\frac{t-t_A}{1+t}\right)^{1/2}+\left(\frac{t_E-t_A}{1+t_E}\right)^{1/2}}{\left(\frac{t-t_A}{1+t}\right)^{1/2}-\left(\frac{t_E-t_A}{1+t_E}\right)^{1/2}}\right]-\ell n\left[\frac{\left(\frac{1+t}{t-t_A}\right)^{1/2}+\left(\frac{1+t_D}{t_D-t_A}\right)^{1/2}}{\left(\frac{1+t}{t-t_A}\right)^{1/2}-\left(\frac{1+t_D}{t_D-t_A}\right)^{1/2}}\right]\right\}$$

$$+\ \alpha - \left\{\pi-2\tan^{-1}\left[\frac{t-t_A}{t_A(1+t)}\right]^{1/2}\right\} + G_{3c}(t) + G_{3s}(t) \qquad (44)$$

$$G_{3c}(t) = -\frac{t}{2\pi}\left[\left(1+\frac{1}{t}\right)\left(1-\frac{t_A}{t}\right)\right]^{1/2}\left\{\int_{t_A}^{t_E} \frac{\ell n\left[1-\frac{2}{F^2}\,\frac{1}{1+\sigma}\,\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)\left[(1+\eta)(\eta-t_A)\right]^{1/2}}\right.$$

$$\left. -\int_{t_D}^{-1} \frac{\ell n\left[1-\frac{2}{F^2}\,\frac{1}{1+\sigma}\,\frac{y_c(\eta)}{\ell}\right]d\eta}{(\eta-t)\left[(1+\eta)(\eta-t_A)\right]^{1/2}}\right\} \qquad (45)$$

and

$$G_{3s}(t) = -\frac{t}{2\pi}\left[\left(1+\frac{1}{t}\right)\left(1-\frac{t_A}{t}\right)\right]^{1/2} \fint_{\beta}^{t_u} \frac{\ell n\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]d\eta}{(\eta-t)\left[(1+\eta)(\eta-t_A)\right]^{1/2}} \qquad (46)$$

Explanation of the slash symbol follows Eq. 41. One will recall that the appropriate t-ranges are $t_E<t<\beta$ for the upper wake boundary, $t<t_D$ for the lower wake boundary, and $t>\beta$ for the free surface.

By using the mathematical definitions given in Eqs. 30-46, it is a somewhat lengthy but straightforward process to develop the final results which follow. First the plate length ℓ can be found by combining Eqs. 10, 12, 13, and 30-35 to get

$$\ell = I_p(-1,t_A) \qquad (47)$$

with

$$I_p(\eta_1,\eta_2) = \frac{2}{(1+t_A)\left[1-\beta\,\ell n(1+1/\beta)\right]}\int_{\eta_1}^{\eta_2}\left\{t_A-\left(\frac{1-t_A}{2}\right)t+\left[t_A(t_A-t)(1+t)\right]^{1/2}\right\}$$

$$\times \exp\left\{-\left[S_1(t)+G_{1c}(t)+G_{1s}(t)\right]\right\}\frac{dt}{\beta-t} \qquad (48)$$

The coordinates of the plate stagnation point are

$$z_B = x_B + iy_B = (\cos\alpha - i\sin\alpha) I_p(0,t_A) \tag{49}$$

and the coordinates of the downstream end of the plate are

$$z_C = x_C + iy_C = (\cos\alpha - i\sin\alpha)\ell \tag{50}$$

It has previously been shown (7) that lift L and drag D are related to the pressure distribution on the plate by the expression

$$D + iL = -i\int_C^A (p-p_c)dz \tag{51}$$

If the lift and drag coefficients, defined as

$$C_L = \frac{L}{\frac{1}{2}\rho q_o^2 \ell} \quad \text{and} \quad C_D = \frac{D}{\frac{1}{2}\rho q_o^2 \ell} \tag{52}$$

are introduced into Eq. 51, then use of Eqs. 1-4 and 8-12 lead to

$$C_D + iC_L = -\frac{i}{\ell}\int_C^A \left[1+\sigma-\frac{2}{F^2}\,\frac{y}{\ell}-e^{\omega+\bar{\omega}} \right] e^{-\omega}\,\frac{dW}{dt}\,dt \tag{53}$$

and ultimately to the results

$$C_D = [1+\sigma + \sin\alpha/F^2 + I_c]\sin\alpha \tag{54}$$

and

$$C_L = [1+\sigma + \sin\alpha/F^2 + I_c]\cos\alpha \tag{55}$$

where

$$I_c = \frac{2}{\ell(1+t_A)[1-\beta\ell n(1+1/\beta)]}\int_{-1}^{t_A} \exp[S_1(t)+G_{1c}(t)+G_{1s}(t)]$$

$$\times\left\{-t_A+\left(\frac{1-t_A}{2}\right)t + [t_A(1+t)(t_A-t)]^{1/2}\right\}\frac{dt}{\beta-t}\,. \tag{56}$$

Substitution of the appropriate expressions for $\omega(t)$ from this section into Eq. 10 will then give the cavity-wake-free surface configurations. The results are the following for the cavity shape:

$$x(t)-x_o = \frac{1}{[1-\beta\ell n(1+1/\beta)](1+\sigma)^{1/2}}\int_{t_o}^{t} \frac{\cos\omega_I(\eta)\ \eta d\eta}{\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]^{1/2}(\beta-\eta)} \tag{57}$$

and

$$y(t)-y_o = \frac{-1}{[1-\beta\ell n(1+1/\beta)](1+\sigma)^{1/2}}\int_{t_o}^{t} \frac{\sin\omega_I(\eta)\ \eta d\eta}{\left[1-\frac{2}{F^2}\frac{1}{1+\sigma}\frac{y_c(\eta)}{\ell}\right]^{1/2}(\beta-\eta)} \tag{58}$$

For the upper boundary select $x_o=0$, $y_o=0$, $t_o=t_A$ and $t_A\leq t<t_E$. For the lower boundary select $x_o=x_c$, $y_o=y_c$, $t=-1$ and $t_D<t\leq-1$. Eq. 38 defines ω_I.

In a very similar way one derives expressions for the wakes and free surface:

$$x(t)-x_o = \frac{1}{1-\beta\ell n(1+1/\beta)}\int_{t_o}^{t} \frac{\cos\gamma(\eta)\ \eta d\eta}{\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]^{m/2}(\beta-\eta)} \tag{59}$$

$$y(t)-y_o = \frac{-1}{1-\beta\ell n(1+1/\beta)}\int_{t_o}^{t} \frac{\sin\gamma(\eta)\ \eta d\eta}{\left[1+\frac{2}{F^2}\left(\frac{d-y_s(\eta)}{\ell}\right)\right]^{m/2}(\beta-\eta)} \tag{60}$$

If $\eta>\beta$, then $m=1$; otherwise $m=0$. For the upper wake boundary $x_o=x_E$, $y_o=y_E$, $t_o=t_E^+$ and $t_E<t<\beta$. For the lower wake $x_o=x_D$, $y_o=y_D$, $t_o=t_D^-$ and $t<t_D$. For the free surface it is not convenient, or even possible, to begin integration at the point $t=\beta$, for this point lies at downstream infinity in the physical plane. As one crosses the point $t=\beta$, there is a finite jump in the y-coordinate by the amount ψ_o to go from the upper wake to the free surface. However, this information by itself is not particularly useful for numerical computation. The next section will clarify this point.

COMPUTATIONAL PROCEDURES AND PROGRAMS

Two FORTRAN IV computer programs have been developed to evaluate the preceding equations. Both program listings will be found in the Appendix. The first, and smaller, program assists one in selecting an appropriate initial set of t-plane parameters or coefficients which relate approximately correctly to the parameter set (ψ_0, α, σ) which one desires to study. The larger program completely evaluates the theory for a given set of t-plane parameters (t_A, t_D, t_E, β). The computational sequence and the role of each major segment within the program and also the required data cards will be described. This section also includes short discussions of some topics the user ought to consider further.

The first program is small and contains approximately 100 statements. In the selection of a coefficient or parameter set for a particular problem, one would like to specify (ψ_0, α, σ) and then use the appropriate corresponding set of t-plane parameters. Due to the mathematical structure of Eqs. 22 and 23, however, it is more appropriate to select ψ_0 or β, and arrays of values for t_A and t_D, and then note the resulting values of β or ψ_0, t_E, α and σ. Moreover, this search must initially be conducted without consideration of gravity effects, for the gravity terms, Eqs. 28 and 29, cannot be evaluated at this early stage of computation. Thus this program performs steps 1 and 2 which are listed following Eq. 29.

The first program operates in the following manner. First either ψ_0 or β is read, and the other parameter is computed from Eq. 13. Then arrays of values for t_A and t_D are read,

t_E is computed by the REAL FUNCTION FE(TD,BETA) which solves Eq. 14 for each t_D, and Eqs. 22 and 23 are then solved for α and σ. Finally, lists of parameter sets are printed.

To process one set of parameters with this program, two data cards are required. The first one has the FORMAT(2F17.7) and supplies either ψ_o or β to the program in that order; supply only one of these two numbers and leave the other space blank. The second data card follows the FORMAT(6F12.7) and contains these data:

TAI, TAINC, TAF, TDI, TDINC, TDF

The program processes an array of t_A values, beginning with the smallest value TAI and proceeding to increase t_A by increments TAINC (>0) until TAF is exceeded. For each value of t_A the program processes an array of t_D values, beginning with the least negative value TDI and decreasing in increments TDINC (<0) until the most negative value TDF has been passed. Then control passes to the head of the program, and another pair of data cards is to be accepted and processed. Alternatively, if one wishes to terminate the program, place one blank card at the end of the other pairs of data cards. Although the Appendix listings show no data cards, both the actual card decks supplied under this project do contain a sample set of data cards.

The large computer program evaluates the equations given earlier and contains almost 1600 statements. The solution for the gravity-affected flow is found by an iterative process. The initial pass through the first portion of the main program computes the physical plane configuration and the force coeffi-

cients in the absence of gravity; values for $y_c(t)$ and $y_s(t)$, σ and α are part of this output. Although these values are altered somewhat when gravity effects are included in the equations, they serve as reasonable first approximations to the later results which are obtained when gravity is considered. When these values do not change significantly from one gravity iteration to the next, the gravity solution has been obtained, and computations should cease. The latter, and larger part, of the main program uses the data for $y_c(t)$, $y_s(t)$, etc. to compute the various gravity terms, such as $G_{1c}(t)$, $G_{1s}(t)$, and so on. Then a new computation of the flow parameters, physical plane configuration and force coefficients begins with the inclusion of the gravity terms that have just been computed. As even one gravity iteration requires a fair amount of computer time, the program is currently designed to calculate only one complete solution and then stop.

To compute one complete case, the program requires three cards of input data. The first card format is FORMAT(4E17.7), and the required data are

TA, TD, TE, BETA

It is assumed that these data form a compatible set and already satisfy Eq. 14. The second data card has the FORMAT(4I5); the values to be read are

NLC, NUC, NFS, IGMAX

Here NLC is the Number of coordinate points on the Lower Cavity boundary which are to be calculated, NUC is the Number of coordinate points on the Upper Cavity boundary, and NFS is the Number of coordinate points to be calculated along the Free

Surface. Minimum values for these control parameters are specified in the program listing, but to achieve some sort of reasonable balance between accuracy of solution and required computation time, it is suggested that these three parameters each be set at approximately 20 to 25. IGMAX is the number of gravity iterations that the program is to compute; if one sets IGMAX = 0, then only the nongravity solution, Ref. (8), is computed. The third data card follows the same format as the first card and inputs values for the two parameters F^2 and t_u, which in the program are called

FSQ, TU

The reading of these three data cards is the first task completed in the computational routine of the program. Arrays of t-plane values along the cavity boundaries are then set up.

Computation of the solution now begins after label 120 is passed. First σ and α are computed, then the plate length ℓ (PL), coordinates of points A, B, and C on the plate are located, and C_L and C_D are computed. Next to be computed are coordinates along the lower cavity boundary, lower wake boundary, upper cavity boundary, and upper wake boundary.

Coordinates along the free surface should be computed now. As the discussion following Eq. 60 pointed out, one cannot numerically integrate through the point $t = \beta$ to get onto the free surface. Instead the program integrates around $t = \beta$ along the arc $t = \beta + \delta\exp(i\theta)$, where $\theta = \pi$ is on the upper wake and $\theta = 0$ is on the free surface. Equation 10 gives the relation between z_w on the wake and z_s on the free surface as

$$z_s - z_w = \frac{i}{[1-\beta \ln(1+1/\beta)]} \int_0^{\pi} (\beta+\delta e^{i\theta}) \exp[-\omega(\beta+\delta e^{i\theta})] d\theta \quad (61)$$

To avoid certain difficulties, the program sets $\delta = (\beta - t_E)/2$. The algebra required to separate Eq. 61 into real and imaginary parts to obtain expressions for the x and y coordinates is sufficiently long that it is not reproduced in the body of this report. However, all the details are programmed in subroutines ARGZ, XARC, YARC, and ARC and some additional subroutines which later supply the gravity-term contributions. Once integration along the arc from the upper wake to the free surface has been achieved, the free surface coordinates are quickly computed, and the initial computation cycle is complete.

At this time three y-coordinate arrays are updated, and the upstream depth d is assigned a value. This particular arrangement assures that $y_c(t)$ and $y_s(t)$, which are used in computing the gravity terms G_{1c}, G_{1s} to G_{3c}, G_{3s} and the gravity terms for the arc and are used in computing the new physical plane configuration, remain unchanged throughout an entire computation cycle. The effect of altering the program to use new values of $y_c(t)$ and $y_s(t)$ immediately is perhaps one of the less important factors that could be investigated while the program is used. One additional point should be noted here about the upstream depth d. Since the wake of the double-spiral-vortex model is presumed to close at downstream infinity, $y_s(\beta)$ is equated to d for subsequent use in evaluating integrals over the range (β, t_u).

At this point in the program a counter IG is incremented

by one and compared with IGMAX; if IG exceeds IGMAX, computation ceases. Early experience in running the program should show whether the iterative approach generates a convergent solution, as was obtained in Ref. (14). Once convergence has been demonstrated, one may wish to replace this program termination feature with a more advanced checking procedure which compares the values of selected parameters from one solution cycle to the next and ends computation when changes become sufficiently small. Parameters that might be used in such a convergence check include α, σ, C_L or C_D and/or the location of selected points in the physical plane.

The remainder of the main program computes arrays of values for the several gravity terms. These arrays are converted into continuous functions by the use of simple, linear interpolation routines. Higher order interpolation schemes were considered but discarded for several reasons. Linear interpolation is simplest and therefore the fastest; if the function being approximated varies slowly or is small, linear routines should be adequate. If the function varies rapidly, then the appropriate control parameter (NLC, NUC, or NFS) should be increased to achieve a better representation. Upon completion of the computation for these gravity terms, control passes back to label 120, and the gravity solution itself is computed. This computation phase now includes the terms B_{1G} and B_{2G} and all the other effects of gravity on the solution.

Based on very limited experience gained during the development and testing of the program, it appears that the inclusion of the gravity terms causes α to increase and σ to decrease

from their former nongravity values for the same t-plane parameters. One should accumulate further experience here and use it in selecting more appropriately the input parameters to the program.

The magnitude of the gravity effect in this problem can be assessed by comparing gravity and nongravity solutions for the same α and σ. The appropriate gravity case should be computed first, using the current computer program. Then the small program should be used to search for the correct set of t-plane parameters which give a match for α and σ. Finally, these parameters should be used in the big program with IGMAX = 0 to obtain the equivalent nongravity solution.

Another point of interest is that it may be feasible to generate a finite cavity solution with $\sigma = 0$ in the presence of gravity, although it has not actually been attempted. The reader is referred to additional remarks on this idea in Ref. (14), p. 335.

Two final points are deserving of some attention. The current program has values for t_u and F^2 assigned from an input data card. It is assumed that the solution is insensitive to the particular value of t_u so long as t_u is sufficiently large; this assumption should be checked by computing several test cases in which only t_u is varied between cases. Program users may also wish to consider the possibility of internally computing a value for F^2.

REFERENCES

1. Wu, T.Y., "A Free Streamline Theory for Two-dimensional Fully Cavitated Hydrofoils," J. Math and Physics, 35, 3, 1956.

2. Wu, T.Y., "A Wake Model for Free-Streamline Theory, Part I," J. Fluid Mech., 13, 2, 1962.

3. Ho, H.T., "Linearized Theory for a Supercavitating Hydrofoil Operating in a Fluid of Finite Depth," Tech. Rept. 119-7, Hydronautics, Laurel, Md., June 1963.

4. Ai, D.K., Acosta, A.J., and Harrison, Z.L., "Linearized Theory of a Two-Dimensional Planing Flat-Plate in a Channel of Finite Depth--I," CIT Hydro. Lab. Rept. No. E-110.2, Pasadena, Calif., April 1964.

5. Yim, B., "On a Fully Cavitating Two-Dimensional Flat-Plate Hydrofoil with Non-Zero Cavitation Number near a Free Surface," Tech. Rept. 463-4, Hydronautics, Laurel, Md., June 1964.

6. Tulin, M.P., "Supercavitating Flows--Small Perturbation Theory," J. Ship Res., 7, 3, 1964.

7. Larock, B.E., and Street, R.L., "A Riemann-Hilbert Problem for Nonlinear, Fully Cavitating Flow," J. Ship Res., 9, 3, December 1965.

8. Larock, B.E., and Street, R.L., "A Nonlinear Solution for a Fully Cavitating Hydrofoil Beneath a Free Surface," J. Ship Res., 11, 2, June 1967.

9. Dinh, N.N., "Some Experiments on a Supercavitating Plane Hydrofoil with Jet-flap," J. Ship Res., 13, 3, September 1969.

10. Parkin, B.R., "A Note on the Cavity Flow Past a Hydrofoil in a Liquid with Gravity," Rept. No. 47-9, Engineering Division, California Institute of Technology, Pasadena, December 1957.

11. Street, R.L., "Supercavitating Flow about a Slender Wedge in a Transverse Gravity Field," J. Ship Res., 7, June 1963.

12. Acosta, A.J., "The Effects of a Longitudinal Gravitational Field on the Supercavitating Flow over a Wedge," J. Applied Mech., 28, Trans. ASME, 83, Series E, 1961.

13. Lenau, C.W., and Street, R.L., "A Non-Linear Theory for Symmetric, Supercavitating Flow in a Gravity Field," J. Fluid Mech., 21, 2, 1965.

14. Larock, B.E., and Street, R.L., "A Nonlinear Theory for a Fully Cavitating Hydrofoil in a Transverse Gravity Field," J. Fluid Mech., 29, 2, August 1967.

APPENDIX -- COMPUTER PROGRAM LISTINGS

Given here is a complete listing of the two FORTRAN IV computer programs described in the body of this report.

```
$SET SINGLE BCD
FILE  5=INPUT, UNIT= READER
FILE  6=LAROCK,UNIT= PRINTER
C
C     GRAVITY EFFECTS ON A CAVITATING FOIL BELOW A FREE SURFACE
C     LAROCK THEORY 1972
C     THIS PROGRAM IS TO BE USED FOR PRELIMINARY COEFFICIENT SELECTION
C
  600 FORMAT(1H0,20X,78H GRAVITY EFFECTS ON A CAVITATING FOIL BELOW A FR
     1EE SURFACE, LAROCK THEORY 1972,//,37X,43H PARAMETER ARRAYS WHICH F
     2OLLOW ARE FOR G=0,///)
      WRITE (6, 600)
C
C     RELATE PSIZERO = H TO BETA, IF BETA IS GIVEN, H IS COMPUTED, IF H
C     IS GIVEN, BETA IS COMPUTED, ASSUMING IT LIES BETWEEN 0.0001 AND
C     10000.0
C
      PI = 3.141592654
   22 READ (5, 610) H, BETA
      IF((H.EQ.0.0).AND.(BETA.EQ.0.0)) GO TO 999
  610 FORMAT(2F17.7)
      IF(H.EQ.0.0) GO TO 11
      BL = 0.0001
      BU = 10000.0
      FL = PI*BL/(1.0-BL*ALOG(1.0+1.0/BL))
      FU = PI*BU/(1.0-BU*ALOG(1.0+1.0/BU))
   12 BM = 0.5*(BL+BU)
      FM = PI*BM/(1.0-BM*ALOG(1.0+1.0/BM))
      IF(FM-H) 13,14,15
   13 FL = FM
      BL = BM
      GO TO 16
   14 BETA = BM
      GO TO 17
   15 FU = FM
      BU = BM
   16 IF((BU-BL).GT.0.0000001) GO TO 12
      BETA = 0.5*(BU+BL)
      GO TO 17
   11 H = PI*BETA/(1.0-BETA*ALOG(1.0+1.0/BETA))
   17 WRITE (6, 611) H, BETA
  611 FORMAT(1H0,30X,11H PSIZERO = ,E17.7,13H  AND  BETA =, E17.7)
C
C     FIND SIGMA AND ALPHA=AL FOR A RANGE OF TA AND TD FOR  G=0.0
C
      WRITE (6, 614)
      READ (5, 612) TAI, TAINC, TAF, TDI, TDINC, TDF
  612 FORMAT(6F12.7)
      TA = TAI
   21 TD = TDI
   20 TE = FE(TD, BETA)
      R1 = SQRT((BETA-TA)/(1.0+BETA))
      R2 = SQRT((TE-TA)/(1.0+TE))
      R3 = 1.0/R1
      R4 = SQRT((1.0+TD)/(TD-TA))
      A11 = (ALOG((R1+R2)/(R1-R2))-ALOG((R3+R4)/(R3-R4)))/(2.0*PI)
```

```
      B1 = 0.5*PI - ARSIN((BETA*(1.0-TA)-2.0*TA)/(BETA*(1.0+TA)))
      R5 = SQRT(1.0+TE) + SQRT(TE-TA)
      R6 = SQRT(TA-TD) + SQRT(-1.0-TD)
      A21 = ALOG(R5/R6)/PI
      B2 = 0.5*PI - ARSIN((1.0-TA)/(1.0+TA))
      REAL LNSIG
      LNSIG = (B1 - B2)/(A11-A21)
      AL = B1 - A11*LNSIG
      SIGMA = EXP(LNSIG) - 1.0
      ALD = AL*180.0/PI
      WRITE (6, 613) SIGMA,ALD,AL,TA,TD,TE,BETA
  613 FORMAT(7E17.7,/)
  614 FORMAT(1H0,6X,6H SIGMA,8X,11H ALPHA,DEG.,6X,11H ALPHA,RAD.,9X,
     13H TA,14X,3H TD,14X,3H TE,13X, 5H BETA,/)
C     PROGRAM ASSUMES  TDI.GT.TDF  AND  TAI.LT.TAF
      IF(TDF.GE.TD)  GO TO 18
      TD = TD + TDINC
      GO TO 20
   18 IF(TA.GE.TAF)  GO TO 22
      TA = TA + TAINC
      GO TO 21
  999 CONTINUE
      STOP
      END
```

REENTRANT FORI
NONREENTRAI

###

```
      REAL FUNCTION FE(TD, BETA)
C     THIS FUNCTION FINDS  TF,  GIVEN  TD  AND  BETA
C     TD.LT.(-1.0), TA.LT.TE.LT.BETA
      C = TD/BETA + ALOG(1.0-TD/BETA)
      ZL = 0.0
      ZU = 0.99
      FL = 0.0
      FU = ZU + ALOG(1.0-ZU)
      IF(FU-C) 5,6,7
    6 FE = ZU*BETA
      RETURN
    7 FL = FU
      ZL = ZU
      ZU = 0.9999999
      FU = ZU + ALOG(1.0-ZU)
    5 ZM = 0.5*(ZU + ZL)
      FM = ZM + ALOG(1.0-ZM)
      IF(FM-C) 8,9,10
    9 FE = FM*BETA
      RETURN
    8 ZU = ZM
      FU = FM
      GO TO 11
   10 ZL = ZM
      FL = FM
   11 IF((ZU-ZL).GT.0.0000001) GO TO 5
      FE = 0.5*(ZL+ZU)*BETA
      RETURN
      END
```

```
###########################################################################

NUMBER OF ERRORS DETECTED = 0000.  NUMBER OF CARDS = 00110.
COMPILATION TIME = 00019 SECONDS ELAPSED.  00002.15 SECONDS PROCESSIN
D2 STACK SIZE = 00007 WORDS.  FILESIZE = 00140 WORDS.
TOTAL PROGRAM CODE = 00347 WORDS.  ARRAY STORAGE = 00049 WORDS.
NUMBER OF PROGRAM SEGMENTS = 0009.  NUMBER OF DISK SEGMENTS = 00001.
ESTIMATED CORE STORAGE REQUIREMENT = 000000 WORDS.
```

```
$SET SINGLE BCD
FILE  5=INPUT, UNIT= READER
FILE  6=LAROCK,UNIT= PRINTER
  601 FORMAT(1H0,20X,78H GRAVITY EFFECTS ON A CAVITATING FOIL BELOW A FR
     1EE SURFACE, LAROCK THEORY 1972,//)
  602 FORMAT(4E17.7)
  603 FORMAT(1H ,10X,11H PSIZERO = , E17.7,/)
  604 FORMAT(1H ,30H INTEGRAL STEP SIZE.LT.MINIMUM)
  605 FORMAT(1H0,10X,15H PLATE LENGTH =, F10.4,/)
  606 FORMAT(1H0,10X, 6H XA = , E16.7,10X,6H YA = , E16.7,/,
     1            11X, 6H XB = , E16.7,10X,6H YB = , E16.7,/,
     2            11X, 6H XC = , E16.7,10X,6H YC = , E16.7,//)
  607 FORMAT(1H0,10X, 5H CL =, E16.7,10X, 5H CD =, E16.7,//)
  608 FORMAT(1H0,10X,29H LOWER STREAMLINE COORDINATES,/)
  609 FORMAT(1H ,17X,2H T,15X,2H X,15X,2H Y,//)
  610 FORMAT(4I5)
  613 FORMAT(7E17.7,/)
  614 FORMAT(1H0,6X,6H SIGMA,8X,11H ALPHA,DEG.,6X,11H ALPHA,RAD.,9X,
     13H TA,14X,3H TD,14X,3H TE,13X, 5H BETA,/)
  615 FORMAT(1H ,5X, 5H PT C, 3E17.7)
  616 FORMAT(1H ,10X, 3E17.7)
  617 FORMAT(1H ,5X, 5H PT D, 3E17.7)
  618 FORMAT(1H0,10X,23H LOWER WAKE COORDINATES,/)
  619 FORMAT(1H0,10X,29H UPPER STREAMLINE COORDINATES,/)
  620 FORMAT(1H , 5X, 5H PT A, 3E17.7)
  621 FORMAT(1H ,5X, 5H PT E, 3E17.7)
  622 FORMAT(1H0,10X,23H UPPER WAKE COORDINATES,/)
  623 FORMAT(1H0,10X,15HARC COORDINATES,//,17X,6H THETA,11X,2H X,15X,
     12H Y,//)
  624 FORMAT(1H0, 25H FREE SURFACE COORDINATES,/)
  625 FORMAT(1H0, 15H GRAVITY ARRAYS,//)
  626 FORMAT(1H , 2E17.7)
  627 FORMAT(1H , 4H G1C,//, 7X,2H T,15X, 7H G1C(T),//)
  628 FORMAT(1H0, 4H G1S,//,7X,2H T,15X, 7H G1S(T),//)
  629 FORMAT(1H0,14H G2C  AND  G2S,//, 9X,2H T,13X,7H G2C(T),10X,7H G2S(
     1T),//)
  630 FORMAT(1H , 3E17.7)
  631 FORMAT(//)
  633 FORMAT(1H0,14H ON LOWER WAKE,//, 7X,2H T,13X,7H G3C(T),10X,7H G3S(
     1T),//)
  634 FORMAT(1H0,14H ON UPPER WAKE,//, 7X,2H T,13X,7H G3C(T),10X,7H G3S(
     1T),//)
  635 FORMAT(1H0,16H ON FREE SURFACE,//, 7X, 2H T,13X,7H G3C(T),10X,7H G
     13S(T),//)
  636 FORMAT(1H0,21H ON ARC AROUND T=BETA,//,17X,6H THETA,11X,5H G3CR,12
     1X,5H G3CI,12X,5H G3SR,12X,5H G3SI,//)
  637 FORMAT(1H , 10X, 5F17.7)
  638 FORMAT(1H0,44H COMMENCE GRAVITY SOLUTION, ITERATION NUMBER, I3,//)
      EXTERNAL F1,F2,F3,F4,F5,F6
      EXTERNAL F7,F8
      EXTERNAL F9,F10,F11,F12
      EXTERNAL F13, F14
      EXTERNAL XARC, YARC
      REAL IC
      REAL LNSIG
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
```

```
      COMMON /C1/ DEL, R2, R4
      COMMON /C2/ TLC, TUC, YLC, YUC
      DIMENSION TLC(100),TUC(100),YLC(100),YUC(100)
      COMMON /C3/ ENLC,ENUC,ENFS,NLC,NUC,NFS
      COMMON /C4/ TFS, YFS
      DIMENSION TFS(100),YFS(100)
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      COMMON /C6/ TCF, GCF, GSF
      DIMENSION TCF(11),GCF(11),GSF(11)
      COMMON /C7/ G2CU,G2SU,G2CL,G2SL
      DIMENSION G2CU(100),G2SU(100),G2CL(100),G2SL(100)
      COMMON /C8/ WL,WU,GCWU,GSWU,GCWL,GSWL
      DIMENSION WL(20),WU(10),GCWU(10),GSWU(10),GCWL(20),GSWL(20)
      COMMON /C9/ JWLM
      COMMON /C10/ GCFS, GSFS
      DIMENSION GCFS(100), GSFS(100)
      COMMON /C11/ GIR,GII,GKR,GKI,THE
      DIMENSION GIR(6),GII(6),GKR(6),GKI(6),THE(6)
      DIMENSION GYLC(100),GYUC(100),GYFS(100)
      DIMENSION XTEM(5), YTEM(5)
      EPSL = 1.0E-06
      EPS = 0.00001
      EPSG = 0.0001
      IG = 0
      WRITE (6, 601)
C
C     FOR G=0 READ  TA, TD, TE, BETA
C
      READ (5, 602)  TA, TD, TE, BETA
C     NLC.GT.2, NUC.GT.7, NFS.GT.6 FOR PROGRAM TO WORK PROPERLY
      READ (5, 610) NLC, NUC, NFS, IGMAX
      READ (5, 602) FSQ, TU
      PI = 3.141592654
      STA = SQRT(TA)
      H = PI*BETA/(1.0-BETA*ALOG(1.0+1.0/BETA))
      WRITE (6, 603) H
C
C     SET UP T-ARRAYS ON CAVITY BOUNDARIES
C
      ENLC = NLC - 1
      ENUC = NUC - 1
      ENFS = NFS - 1
      TLC(1) = -1.0
      TINC = (1.0+TD)/ENLC
      DO 2 J=2,NLC
    2 TLC(J) = TLC(J-1) + TINC
      TLC(NLC) = TLC(NLC) + EPSL
      TINC = 0.4*TA
      TUC(1) = TA
      DO 3 J=2,6
    3 TUC(J) = TUC(J-1) + TINC
      TINC = (TE-3.0*TA)/(ENUC-5.0)
      DO 4 J=7,NUC
    4 TUC(J) = TUC(J-1) + TINC
      TUC(NUC) = TUC(NUC) - EPSL
C
C     FIND SIGMA AND ALPHA=AL
C
  120 CONTINUE
      IF(IG.GT.0) WRITE (6, 638) IG
```

```
      WRITE (6, 614)
      R1 = SQRT((BETA-TA)/(1.0+BETA))
      R2 = SQRT((TE-TA)/(1.0+TE))
      R3 = 1.0/R1
      R4 = SQRT((1.0+TD)/(TD-TA))
      A11 = (ALOG((R1+R2)/(R1-R2))-ALOG((R3+R4)/(R3-R4)))/(2.0*PI)
      B1 = 0.5*PI - ARSIN((BETA*(1.0-TA)-2.0*TA)/(BETA*(1.0+TA)))
      R5 = SQRT(1.0+TE) + SQRT(TE-TA)
      R6 = SQRT(TA-TD) + SQRT(-1.0-TD)
      A21 = ALOG(R5/R6)/PI
      B2 = 0.5*PI - ARSIN((1.0-TA)/(1.0+TA))
      IF (IG.EQ.0) GO TO 10
C     NOW COMPUTE GRAVITY CONTRIBUTIONS TO  B1  AND  B2
      B1G = -(GCWU(8)+GSWU(8))+3.0*(GCWU(9)+GSWU(9)-GCWU(10)-GSWU(10))
      B1 = B1 + B1G
      A1 = TA+EPSG
      A2 = A1+25.0*EPSG
      A3 = TUC(NUC)
      A4 = TLC(NLC)
      A6 = -1.0-EPSG
      A5 = A6-25.0*EPSG
      SMIN1 = EPSG/105.0
      SMIN2 = (A3-A2)/2100.0
      SMIN3 = (A5-A4)/2100.0
      I = NEWCO(A1, A2, F13, RES1, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A3, F13, RES2, EPSG, SMIN2)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A4, A5, F13, RES3, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A5, A6, F13, RES4, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      B2G = RES1+RES2-RES3-RES4 -GZ
      AA = BETA
      DO 11 KK=1,5
      FK = KK
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      SMIN = (BB-AA)/2100.0
      I = NEWCO(AA, BB, F14, RES, EPSG, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      B2G = B2G + RES
      AA = BB
   11 CONTINUE
      B2 = B2 - 0.5*B2G/PI
   10 CONTINUE
C     SOLVE PARAMETER EQUATIONS HERE
      LNSIG = (B1 - B2)/(A11-A21)
      AL = B1 - A11*LNSIG
      SIGMA = EXP(LNSIG) - 1.0
      ALD = AL*180.0/PI
      WRITE (6, 613) SIGMA,ALD,AL,TA,TD,TE,BETA
      FACS = (1.0-BETA*ALOG(1.0+1.0/BETA))*SQRT(1.0+SIGMA)
      FACW = 1.0-BETA*ALOG(1.0+1.0/BETA)
C
C     FIND FOIL COORDINATES, CL AND CD
C
      FAC = 2.0/((1.0+TA)*(1.0-BETA*ALOG(1.0+1.0/BETA)))
      SMIN = (1.0+TA)/4200.0
      AA = -1.0 + EPSL
```

```
      BB = TA - FPSL
      I = NEWCO(AA, BB,   F1, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      PL = FAC*RES
      WRITE (6, 605) PL
      SMIN = TA/4200.0
      I = NEWCO(0.0, BB, F1, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      XA = 0.0
      YA = 0.0
      XB = FAC*RES*COS(AL)
      YB =-FAC*RES*SIN(AL)
      XC = PL*COS(AL)
      YCC=-PL*SIN(AL)
      WRITE (6, 606) XA, YA, XB, YB, XC, YCC
      SMIN = (1.0+TA)/4200.0
      I = NEWCO(AA, BB,   F2, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      IC = FAC*RES/PL
      SAF = 0.0
      IF(IG.GT.0)  SAF = SIN(AL)/FSQ
      CL = (1.0+SIGMA  + SAF        + IC)*COS(AL)
      CD = (1.0+SIGMA  + SAF        + IC)*SIN(AL)
      WRITE (6, 607) CL, CD
      FSFAC = 2.0/(FSQ*(1.0+SIGMA)*PL)
C
C     COMPUTE LOWER CAVITY BOUNDARY COORDINATES
C
      WRITE (6, 608)
      WRITE (6, 609)
      X = XC
      Y = YCC
      GYLC(1) = Y
      AA = -1.0
      WRITE (6, 615) AA, X, Y
      DO 36 J=2,NLC
      BB = TLC(J)
      SMIN = (AA-BB)/4200.0
      I = NEWCO(AA, BB, F3, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      X = X + RES/FACS
      I = NEWCO(AA, BB, F4, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      Y = Y + RES/FACS
      GYLC(J) = Y
      IF(J.EQ.NLC) GO TO  35
      WRITE (6, 616) BB, X, Y
      AA = BB
   36 CONTINUE
   35 WRITE (6, 617)  TD, X, Y
C
C     COMPUTE SOME LOWER WAKE BOUNDARY COORDINATES
C
      WRITE (6, 618)
      WRITE (6, 609)
      WRITE (6, 617) TD, X, Y
      XD = X
      FPL = 5.0*PL
      JWL = 1
      WL(JWL) = TD - EPSL
```

```
      AA = TD
      TINC = 1.0+TD
   41 BB = AA+TINC
      IF(AA.EQ.TD) AA=AA-EPSL
      JWL = JWL + 1
      WL(JWL) = BB
      SMIN = (AA-BB)/4200.0
      I = NEWCO(AA, BB, F5, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      XT = RES/FACW
      IF(XT.LT.PL) TINC = 1.5*TINC
      X = X + XT
      I = NEWCO(AA, BB, F6, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      Y = Y + RES/FACW
      WRITE (6, 616) BB, X, Y
      AA = BB
C     CHECK TO END WAKE CALCULATIONS
      IF((X-XD).LT.FPL) GO TO 41
      JWLM = JWL
C
C     COMPUTE UPPER CAVITY BOUNDARY STREAMLINES
C
      X = XA
      Y = YA
      AA = TA
      GYUC(1) = Y
      WRITE (6, 619)
      WRITE (6, 609)
      WRITE (6, 620) AA, X, Y
      DO 46 J=2,NUC
      BB = TUC(J)
      SMIN = (BB-AA)/4200.0
      I = NEWCO(AA, BB, F3, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      X = X + RES/FACS
      I = NEWCO(AA, BB, F4, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      Y = Y + RES/FACS
      GYUC(J) = Y
      IF(J.EQ.NUC) GO TO  45
      WRITE (6, 616) BB, X, Y
      AA = BB
   46 CONTINUE
   45 WRITE (6, 621) TE, X, Y
C
C     COMPUTE SOME UPPER WAKE COORDINATES
C
      WRITE (6, 622)
      WRITE (6, 609)
      WRITE (6, 621) TE, X, Y
      TINC = 0.1*(BETA-TE)
      IF(IG.GT.0) GO TO 147
      WU(1) = TE+EPSL
      WU(2) = TE+TINC
      DO 47 J=3,10
   47 WU(J) = WU(J-1) + TINC
  147 CONTINUE
      AA = WU(1)
      SMIN = (BETA-TE)/4200.0
```

```
      DO 48 J=2,10
      BB = WU(J)
      I = NEWCO(AA, BB, F5, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      X = X + RES/FACW
      I = NEWCO(AA, BB, F6, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      Y = Y + RES/FACW
      IF(J.EQ.6)  XW = X
      IF(J.EQ.6)  YW = Y
      WRITE (6, 616) BB, X, Y
      AA = BB
   48 CONTINUE
C
C     INTEGRATE ALONG ARC TO FREE SURFACE
C
      WRITE (6, 623)
      IF(IG.GT.0) GO TO 151
      DTHET = 0.2*PI
      THE(1) = 0.0
      DO 51 J=2,6
   51 THE(J) = THE(J-1) + DTHET
      DEL = 0.5*(BETA-TE)
  151 CONTINUE
      X = XW
      Y = YW
      WRITE (6, 616) THE(1), X, Y
      SMIN = DTHET/2100.0
      EPS = 10.0*EPS
      DO 52 J=2,6
      AA = THE(J-1)
      BB = THE(J)
      I = NEWCO(AA, BB, XARC, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      X = X + RES/FACW
      I = NEWCO(AA, BB, YARC, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      Y = Y + RES/FACW
   52 WRITE (6, 616)  THE(J), X, Y
      EPS = 0.1*EPS
C
C     COMPUTE FREE SURFACE COORDINATES
C
      IF(IG.GT.0) GO TO 205
      SET UP  T-ARRAY ON FREE SURFACE
      TFS(1) = BETA + TINC
      DO 5 J=2,5
    5 TFS(J) = TFS(J-1) + TINC
      DO 6 J=6,NFS
      ENJ = J
      TFAC = (ENJ-5.0)/(FNFS-4.0)
      TFAC = TFAC**3.0
    6 TFS(J) = TFS(5) + (TU-BETA-DEL)*TFAC
C
  205 CONTINUE
      WRITE (6, 624)
      WRITE (6, 609)
      XTEM(5) = X
      YTEM(5) = Y
      DO 53 J=1,4
```

```
      K = 6 - J
      AA = TFS(K)
      BB = TFS(K-1)
      SMIN = (AA-BB)/2100.0
      I = NEWCO(AA, BB, F5, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      XTEM(K-1) = XTEM(K) + RES/FACW
      I = NEWCO(AA, BB, F6, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      YTEM(K-1) = YTEM(K) + RES/FACW
   53 CONTINUE
      DO 54 J=1,5
      GYFS(J) = YTEM(J)
      WRITE (6, 616) TFS(J), XTEM(J), YTEM(J)
   54 CONTINUE
      DO 55 J=6,NFS
      AA = TFS(J-1)
      BB = TFS(J)
      SMIN = (BB-AA)/2100.0
      I = NEWCO(AA, BB, F5, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      X = X + RES/FACW
      I = NEWCO(AA, BB, F6, RES, EPS, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      Y = Y + RES/FACW
      GYFS(J) = Y
      WRITE (6, 616) BB, X, Y
   55 CONTINUE
C
C     UPDATE FREE SURFACE ARRAY DESCRIPTIONS
C
      DO 60 J=1,NLC
   60 YLC(J) = GYLC(J)
      DO 61 J=1,NUC
   61 YUC(J) = GYUC(J)
      DO 62 J=1,NFS
   62 YFS(J) = GYFS(J)
      D = GYFS(NFS)
C
C     COMPUTATION CYCLE HAS BEEN COMPLETED FOR G=0. FOR G.NE.0,GRAVITY
C     TERMS ARE COMPUTED BELOW AND SOLUTION IS RECOMPUTED, BEGINNING AT
C     LABEL 120. CHECK IS MADE HERE TO SEE IF NO. OF GRAVITY ITERATIONS
C     EQUALS IGMAX, AT WHICH TIME COMPUTATION CEASES.
C
      IG = IG + 1
      IF(IG.GT.IGMAX) GO TO 999
C
C     COMPUTE GCF AND GSF ARRAYS, WHICH LATER ARE INTERPOLATED AMONG
C     TO GIVE G1C(T) AND G1S(T)
C
      TCFINC = 0.1*(1.0+TA)
      WRITE (6, 625)
      WRITE (6, 627)
      DO 70 J=1,11
      EJ = J-1
      TCF(J) = -1.0 + EJ*TCFINC
   70 CONTINUE
      A1 = TA + EPSG
      A2 = -1.0 - EPSG
      A3 = A1 + 10.0*EPSG
```

```
      A4 = TE - 10.0*EPSG
      SMIN1 = EPSG/105.0
      SMIN2 = (A4-A3)/2100.0
      A5 = TD + 10.0*EPSG
      A6 = A2 - 10.0*EPSG
      SMIN3 = (A6-A5)/2100.0
      GCF(1) = 0.5*ALOG(1.0-YC(-1.0)*FSFAC)
      GCF(11) = 0.0
      WRITE (6, 626) TCF(1), GCF(1)
      GZ = 4.0*SQRT(EPSG/(1.0+TA))*GCF(1)
C     START GCF FOR G1C
      DO 71 J=2,10
      T = TCF(J)
      I = NEWCO(A1, A3, F7, RES1, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A3, A4, F7, RES2, EPSG, SMIN2)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A4, TUC(NUC), F7, RES3, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      RESA = RES1 + RES2 + RES3
      I = NEWCO(TLC(NLC), A5, F7, RES1, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A5, A6, F7, RES2, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A6, A2, F7, RES3, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      RESB = RES1 + RES2 + RES3
      TEMP = RESA - RESB +GZ/(1.0+T)
      QRT = 0.5*SQRT((1.0+T)*(TA-T))/PI
      GCF(J) = TEMP*QRT
      WRITE (6, 626) T, GCF(J)
   71 CONTINUE
      WRITE (6, 626) TCF(11), GCF(11)
C     START GSF FOR G1S
      WRITE (6, 628)
      GSF(1) = 0.0
      GSF(11) = 0.0
      WRITE (6, 626) TCF(1) , GSF(1)
      DO 72 J=2,10
      T = TCF(J)
      AA = BETA
      KK = 0
      BB = BETA + DEL
      SMIN3 = DEL/2100.0
      I = NEWCO(AA, BB, F8, RES3, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      AA = BB
      DO 73 KK=1,5
      FK = KK
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      IF(BB.LE.AA)  GO TO 733
      SMIN3 = (BB-AA)/2100.0
      I = NEWCO(AA, BB, F8, RES, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
      AA = BB
  733 CONTINUE
   73 CONTINUE
      GSF(J) = 0.5*SQRT((1.0+T)*(TA-T))*RES3/PI
```

```
      WRITE (6, 626) TCF(J), GSF(J)
   72 CONTINUE
      WRITE (6, 626) TCF(11), GSF(11)
C
C     COMPUTE CAVITY GRAVITY ARRAYS
C     THESE ARRAYS ARE INTERPOLATED AMONG TO FORM G2C AND G2S
C
C     UPPER CAVITY STREAMLINE
C
      G2CU(1) = 0.0
      G2SU(1) = 0.0
      WRITE (6, 629)
      WRITE (6, 630) TUC(1), G2CU(1), G2SU(1)
      DIMENSION AU(10), FSMN(5)
      DO 75 J=2,NUC
      ES = EPSG
      T = TUC(J)
      IF((T-TA).LT.(4.0*EPSG)) ES = (T-TA)/4.0
      NJI = 3
      IF((J.GT.6).AND.(J.LT.NUC)) NJI = 5
      IF(J.GT.6) GO TO 76
      AU(1) = TA+ES
      AU(2) = T-ES
      ESMN(1) = (AU(2)-AU(1))/1050.0
      AU(3) = T+ES
      AU(4) = TUC(7)
      ESMN(2) = (AU(4)-AU(3))/1050.0
      AU(5) = AU(4)
      AU(6) = TUC(NUC)
      ESMN(3) = (AU(6)-AU(5))/2100.0
      GO TO 77
   76 AU(1) = TA+EPSG
      AU(2) = AU(1) + 10.0*EPSG
      ESMN(1) = EPSG/105.0
      AU(3) = AU(2)
      AU(4) = T - 10.0*EPSG
      ESMN(2) = (AU(4)-AU(3))/2100.0
      AU(5) = AU(4)
      AU(6) = T - EPSG
      ESMN(3) = ESMN(1)
      IF(J.EQ.NUC) GO TO 77
      AU(7) = T + EPSG
      AU(8) = AU(7) + 10.0*EPSG
      ESMN(4) = ESMN(1)
      AU(9) = AU(8)
      AU(10) = TUC(NUC)
      ESMN(5) = (AU(10)-AU(9))/2100.0
   77 RES = 0.0
      DO 771 JJ=1,NJI
      J2 = 2*JJ
      J1 = J2 - 1
      I = NEWCO(AU(J1), AU(J2), F7, RES1, EPSG, ESMN(JJ))
      IF(I.GT.1) WRITE (6, 604)
      RES = RES + RES1
  771 CONTINUE
      A1 = TLC(NLC)
      A2 = -1.0-11.0*EPSG
      A3 = -1.0 - EPSG
      SMIN1 = EPSG/105.0
      SMIN2 = (A2-A1)/2100.0
```

```
      I = NEWCO(A1, A2, F7, RES1, EPSG, SMIN2)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A3, F7, RES2, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      TEMP = RES-RES1-RES2 +GZ/(1.0+T)
      QRT = -0.5*T*SQRT((1.0+1.0/T)*(1.0-TA/T))/PI
      G2CU(J) = QRT*TEMP
      AA = BETA
      KK = 0
      BB = BETA + DEL
      SMIN3 = DEL/2100.0
      I = NEWCO(AA, BB, F8, RES3, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      AA = BB
      DO 772 KK=1,5
      FK = KK
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      IF(BB.LE.AA)  GO TO 773
      SMIN3 = (BB-AA)/2100.0
      I = NEWCO(AA, BB, F8, RES, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
      AA = BB
  773 CONTINUE
  772 CONTINUE
      G2SU(J) = QRT*RES3
   75 CONTINUE
C
C     LOWER CAVITY STREAMLINE
C
      WRITE (6, 631)
      G2CL(1) = 0.0
      G2SL(1) = 0.0
      WRITE (6, 630) TLC(1), G2CL(1), G2SL(1)
      A1 = TA + EPSG
      A2 = A1 + 25.0*EPSG
      SMIN1 = EPSG/105.0
      A3 = TUC(NUC)
      SMIN2 = (A3-A2)/2100.0
      AU(1) = -1.0-EPSG
      AU(2) = AU(1) - 10.0*EFSG
      ESMN(1) = EPSG/105.0
      AU(3) = AU(2)
      ESMN(3) = ESMN(1)
      ESMN(4) = FSMN(1)
      AU(10) = TLC(NLC)
      DO 78 J=2,NLC
      T = TLC(J)
      NJI = 5
      IF(J.EQ.NLC) NJI = 3
      I = NEWCO(A1, A2, F7, RES1, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A3, F7, RES2, EPSG, SMIN2)
      IF(I.GT.1) WRITE (6, 604)
      TEMP = RES1 + RES2
      AU(4) = T + 10.0*EPSG
      ESMN(2) = (AU(3)-AU(4))/2100.0
      AU(5) = AU(4)
      AU(6) = T + EPSG
```

```
      IF(J.EQ.NLC) GO TO 80
      AU(7) = T - EPSG
      AU(8) = AU(7) - 10.0*EPSG
      AU(9) = AU(8)
      ESMN(5) = (AU(9)-AU(10))/2100.0
   80 RES = 0.0
      DO 774 JJ=1,NJT
      J2 = 2*JJ
      J1 = J2 - 1
      I = NEWCO(AU(J1), AU(J2), F7, RES1, EPSG, ESMN(JJ))
      IF(I.GT.1) WRITE (6, 604)
      RES = RES + RES1
  774 CONTINUE
      TEMP = TEMP + RES + GZ/(1.0+T)
      QRT = -0.5*T*SQRT((1.0+1.0/T)*(1.0-TA/T))/PI
      G2CL(J) = QRT*TEMP
      AA = BETA
      KK = 0
      BB = BETA + DEL
      SMIN3 = DEL/2100.0
      I = NEWCO(AA, BB, F8, RES3, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      AA = BB
      DO 775 KK=1,5
      FK = KK
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      IF(BB.LE.AA)  GO TO 755
      SMIN3 = (BB-AA)/2100.0
      I = NEWCO(AA, BB, F8, RES, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
      AA = BB
  755 CONTINUE
  775 CONTINUE
      G2SL(J) = QRT*RES3
      WRITE (6, 630) TLC(J), G2CL(J), G2SL(J)
   78 CONTINUE
C
C     COMPUTE WAKE AND FREE SURFACE GRAVITY ARRAYS
C     THESE ARRAYS ARE INTERPOLATED AMONG TO FORM G3C AND G3S
C
C     LOWER WAKE
C
      WRITE (6, 631)
      A1 = TA+EPSG
      A2 = A1+25.0*EPSG
      A3 = TUC(NUC)
      A4 = TLC(NLC)
      A5 = A4+10.0*EPSG
      A7 = -1.0-EPSG
      A6 = A7-25.0*EPSG
      A8 = A3 - 10.0*EPSG
      SMIN1 = EPSG/105.0
      SMIN2 = (A3-A2)/2100.0
      SMIN3 = (A6-A5)/2100.0
      SMIN4 = (A8-A2)/2100.0
      WRITE (6, 633)
      DO 81 J=1,JWLM
      T = WL(J)
```

```
      I = NEWCO(A1, A2, F7, RES1, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A3, F7, RES2, EPSG, SMIN2)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A4, A5, F7, RES3, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A5, A6, F7, RES4, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A6, A7, F7, RES5, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      TEMP = RES1 + RES2 - RES3 - RES4 - RES5 + GZ/(1.0+T)
      QRT = -0.5*T*SQRT((1.0+1.0/T)*(1.0-TA/T))/PI
      GCWL(J) = TEMP*QRT
      AA = BETA
      KK = 0
      BB = BETA + DEL
      SMIN  = DEL/2100.0
      I = NEWCO(AA, BB, F8, RES3, EPSG, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      AA = BB
      DO 776 KK=1,5
      FK = KK
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      IF(BB.LE.AA)  GO TO 756
      SMIN  = (BB-AA)/2100.0
      I = NEWCO(AA, BB, F8, RES, EPSG, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
      AA = BB
  756 CONTINUE
  776 CONTINUE
      GSWL(J) = RES3*QRT
      WRITE (6, 630) T, GCWL(J), GSWL(J)
   81 CONTINUE
C
C     UPPER WAKE
C
      WRITE (6, 634)
      DO 82 J=1,10
      T = WU(J)
      I = NEWCO(A1, A2, F7, RES1, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A8, F7, RES2, EPSG, SMIN4)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A8, A3, F7, RES3, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A4, A5, F7, RES4, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A5, A6, F7, RES5, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A6, A7, F7, RES6, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      TEMP = RES1+RES2+RES3-RES4-RES5-RES6 + GZ/(1.0+T)
      QRT = -0.5*T*SQRT((1.0+1.0/T)*(1.0-TA/T))/PI
      GCWU(J) = QRT*TEMP
      RES3 = 0.0
      AA = BETA
      KK = 0
      SMIN3 = DEL/2100.0
```

```
   83 BB = AA + DEL
      I = NEWCO(AA, BB, F8, RES,  EPSG, SMIN3)
      RES3 = RES3 + RES
      IF(I.GT.1) WRITE (6, 604)
      KK = KK + 1
      AA = BB
      IF(KK.EQ.1)  GO TO  83
      DO 777 KK=2,6
      FK = KK - 1
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      IF(BB.LE.AA)  GO TO 757
      SMIN3 = (BB-AA)/2100.0
      I = NEWCO(AA, BB, F8, RES, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
      AA = BB
  757 CONTINUE
  777 CONTINUE
      GSWU(J) = RES3*QRT
      WRITE (6, 630) T, GCWU(J), GSWU(J)
   82 CONTINUE
C
C     FREE SURFACE
C
      WRITE (6, 635)
      DO 85 J=1,NFS
      T = TFS(J)
      I = NEWCO(A1, A2, F7, RES1, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A8, F7, RES2, EPSG, SMIN4)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A8, A3, F7, RES3, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A4, A5, F7, RES4, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A5, A6, F7, RES5, EPSG, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A6, A7  F7, RES6, EPSG, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      TEMP = RES1+RES2+RES3-RES4-RES5-RES6 + GZ/(1.0+T)
      QRT = -0.5*T*SQRT((1.0+1.0/T)*(1.0-TA/T))/PI
      GCFS(J) = QRT*TEMP
C     BETWEEN HERE AND LABEL 106 PRIMARILY SETS UP INTEGRATION LIMITS
C     FOR EVALUATING G3S(T), T.GT.BETA
      AA = BETA
      KK = 0
      RES3 = 0.0
      SMIN = DEL/2100.0
   90 BB = AA + DEL
      IF((T.GE.AA).AND.(T.LE.BB)) GO TO 91
      IF((T.LT.AA).AND.((T+EPSG).GT.AA)) AA = T+EPSG
      IF(AA.GE.BB) GO TO 93
      BTEMP = BB
      IF((T.GT.BB).AND.((T-EPSG).LT.BB)) BB = T-EPSG
      IF(BB.LE.AA) GO TO 94
      I = NEWCO(AA, BB, F8, RES, EPSG, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
   94 BB = BTEMP
```

```
      GO TO 93
   91 DD = T-EPSG
      IF(DD.LE.AA) GO TO 92
      SMIN = (DD-AA)/2100.0
      I = NEWCO(AA, DD, F8, RES, EPSG, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
   92 CC = T+EPSG
      IF(CC.GE.BB) GO TO 93
      SMIN = (BB-CC)/2100.0
      I = NEWCO(CC, BB, F8, RES, EPSG, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
   93 KK = KK+1
      AA = BB
      IF(KK.EQ.1) GO TO 90
      DO 778 KK=2,6
      FK = KK-1
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      IF(BB.LE.AA) GO TO 758
      IF((T.GE.AA).AND.(T.LE.BB)) GO TO 95
      IF((T.LT.AA).AND.((T+10.0*EPSG).GT.AA)) GO TO 96
      IF((T.GT.BB).AND.((T-10.0*EPSG).LT.BB)) GO TO 97
      JF = 1
      AU(1) = AA
   99 AU(2) = BB
      GO TO 106
   96 JF = 2
      AU(1) = AA
      IF((T+EPSG).GT.AA) AU(1) = T+EPSG
      AU(2) = T+10.0*EPSG
      IF(AU(2).LT.BB) GO TO 98
      JF = 1
      GO TO 99
   98 AU(3) = AU(2)
      AU(4) = BB
      GO TO 106
   97 JF = 2
      AU(1) = AA
      AU(2) = T-10.0*EPSG
      IF(AU(2).GT.AA) GO TO 100
      JF = 1
      AU(2) = BB
      IF((T-EPSG).LT.BB) AU(2) = T-EPSG
      GO TO 106
  100 AU(3) = AU(2)
      AU(4) = BB
      IF((T-EPSG).LT.BB) AU(4) = T-EPSG
      GO TO 106
   95 TMTE = T - 10.0*EPSG
      TME = T - EPSG
      TPE = T + EPSG
      TPTE = T + 10.0*EPSG
      IF((TMTE.GT.AA).AND.(TPTE.LT.BB)) GO TO 102
      IF(TME.LE.AA) GO TO 103
      IF(TMTE.LE.AA) GO TO 104
      IF(TPE.GE.BB) GO TO 105
      JF = 3
      AU(1) = AA
```

```
      AU(2) = TMTE
      AU(3) = AU(2)
      AU(4) = TME
      AU(5) = TPE
      AU(6) = BB
      GO TO 106
  105 JF = 2
      AU(1) = AA
      AU(2) = TMTE
      AU(3) = AU(2)
      AU(4) = TME
      GO TO 106
  104 JF = 3
      AU(1) = AA
      AU(2) = TME
      AU(3) = TPE
      AU(4) = TPTE
      AU(5) = AU(4)
      AU(6) = BB
      GO TO 106
  103 JF = 2
      AU(1) = TPE
      AU(2) = TPTE
      AU(3) = AU(2)
      AU(4) = BB
      GO TO 106
  102 JF = 4
      AU(1) = AA
      AU(2) = TMTE
      AU(3) = AU(2)
      AU(4) = TME
      AU(5) = TPE
      AU(6) = TPTE
      AU(7) = AU(6)
      AU(8) = BB
  106 CONTINUE
C     LOOP 107 EVALUATES THE INTEGRAL FOR G3S
      DO 107 JJ=1,JF
      J2 = 2*JJ
      J1 = J2 - 1
      ESMN(JJ) = (AU(J2)-AU(J1))/2100.0
      I = NEWCO(AU(J1), AU(J2), F8, RES, EPSG, ESMN(JJ))
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
  107 CONTINUE
      AA = BB
  758 CONTINUE
  778 CONTINUE
      GSFS(J) =RES3*QRT
      WRITE (6, 630) T, GCFS(J), GSFS(J)
   85 CONTINUE
C
C     COMPUTE GRAVITY TERMS ALONG ARC AROUND T=BETA
C
C     THESE TERMS ARE EVALUATED CAREFULLY BUT NOT ESPECIALLY ACCURATELY
      EPS3 = 0.001
      EPS2 = 0.01
      WRITE (6, 636)
      J=1
      T = BETA + DEL
```

```
      GII(J) = 0.0
      GIR(J) = GCFS(5)
      GKR(J) = GSFS(5)
      GKI(J) = -0.5*ALOG(1.0+2.0*(D-YS(T))/(FSQ*PL))
      WRITE (6, 637) THE(J),GIR(J),GII(J),GKR(J),GKI(J)
      TINC = 0.1*(BETA-TE)
      SMIN = TINC/2100.0
      DO 110 J=2,5
      T = THE(J)
      DCT = DEL*COS(T)
      DST = DEL*SIN(T)
      XH = (1.0+BETA+DCT)*(BETA+DCT-TA)-DST*DST
      YH = DST*(1.0-TA+2.0*(BETA+DCT))
      TEMP = SQRT(XH*XH+YH*YH)
      XJ = SQRT(0.5*( XH+TEMP))
      YJ = SQRT(0.5*(-XH+TEMP))
      TEM = 1.0+BETA+DCT
      DNM = TEM*TEM + DST*DST
      COR1 = GZ*TEM/DNM
      COR2 = -GZ*DST/DNM
      I = NEWCO(A1, A2, F9, RES1, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A8, F9, RES2, EPS3, SMIN4)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A8, A3, F9, RES3, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A4, A5, F9, RES4, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A5, A6, F9, RES5, EPS3, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A6, A7, F9, RES6, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      TEMR = RES1+RES2+RES3-RES4-RES5-RES6 + COR1
      I = NEWCO(A1, A2, F10,RES1, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A2, A8, F10,RES2, EPS3, SMIN4)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A8, A3, F10,RES3, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A4, A5, F10,RES4, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A5, A6, F10,RES5, EPS3, SMIN3)
      IF(I.GT.1) WRITE (6, 604)
      I = NEWCO(A6, A7, F10,RES6, EPS3, SMIN1)
      IF(I.GT.1) WRITE (6, 604)
      TEMI = RES1+RES2+RES3-RES4-RES5-RES6 + COR2
      GIR(J) = -0.5*(XJ*TEMR - YJ*TEMI)/PI
      GII(J) = -0.5*(YJ*TEMR + XJ*TEMI)/PI
C     FREE SURFACE TERMS
      RES2 = 0.0
      RES3 = 0.0
      AA = BETA
      DO 781 KK=11,20
      BB = AA + TINC
      I = NEWCO(AA, BB, F11, RES, EPS2, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      RES2 = RES2 + RES
      I = NEWCO(AA, BB, F12, RES, EPS2, SMIN)
      IF(I.GT.1) WRITE (6, 604)
      RES3 = RES3 + RES
```

```
      AA = BB
  781 CONTINUE
      DO 780 KK=2,6
      FK = KK - 1
      FK = 0.04*FK*FK
      BB = BETA + FK*(TU-BETA)
      IF(BB.LE.AA)  GO TO 760
      SMIN3 = (BB-AA)/4200.0
      I = NEWCO(AA, BB, F11, RES, EPS3, SMIN3)
      IF(I.GT.1)  WRITE (6, 604)
      RES2 = RES2 + RES
      I = NEWCO(AA, BB, F12, RES, EPS3, SMIN3)
      IF(I.GT.1)  WRITE (6, 604)
      RES3 = RES3 + RES
      AA = BB
  760 CONTINUE
  780 CONTINUE
      GKR(J) = -0.5*(XJ*RES2 - YJ*RES3)/PI
      GKI(J) = -0.5*(XJ*RES3 + YJ*RES2)/PI
      WRITE (6, 637) THE(J),GIR(J),GII(J),GKR(J),GKI(J)
  110 CONTINUE
      J=6
      GII(J) = 0.0
      GIR(J) = GCWU(6)
      GKR(J) = GSWU(6)
      GKI(J) = 0.0
      WRITE (6, 637) THE(J),GIR(J),GII(J),GKR(J),GKI(J)
C
C     ALL GRAVITY ARRAYS COMPLETE
C
      GO TO 120
C
C
C
  999 CONTINUE
      STOP
      END
```

REENTRANT FOR
NONREENTRA

```
*************************************************************************

      REAL FUNCTION F1(F)
C     THIS FUNCTION COMPUTES THE INTEGRAND FOR THE PLATE LENGTH
      COMMON TA,TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      SA = ((1.0+2.0*TD-TA)*F+(1.0-TA)*TD-2.0*TA)/((F-TD)*(1.0+TA))
      SB = ((1.0+2.0*TF-TA)*F+(1.0-TA)*TF-2.0*TA)/((TF-E)*(1.0+TA))
      S1 = (1.0+(ARSIN(SA)+ARSIN(SB))/PI)*LNSIG*0.5
      IF(IG.GT.0)  S1 = S1 + G1C(F) + G1S(E)
      SA = TA-0.5*E*(1.0-TA)+SQRT(TA*(TA-E)*(1.0+E))
      F1 = SA*EXP(-S1)/(BETA-E)
      RETURN
      END

*************************************************************************
```

```
      REAL FUNCTION F2(E)
C     THIS FUNCTION COMPUTES THE INTEGRAND, CALLED IC, FOR CL, CD
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      SA = ((1.0+2.0*TD-TA)*E+(1.0-TA)*TD-2.0*TA)/((E-TD)*(1.0+TA))
      SB = ((1.0+2.0*TE-TA)*E+(1.0-TA)*TE-2.0*TA)/((TE-E)*(1.0+TA))
      S1 = (1.0+(ARSIN(SA)+ARSIN(SB))/PI)*LNSIG*0.5
      IF(IG.GT.0)  S1 = S1 + G1C(E) + G1S(E)
      SA = -TA+0.5*E*(1.0-TA)+SQRT(TA*(1.0+E)*(TA-E))
      F2 = SA*EXP(S1)/(BETA-E)
      RETURN
      END

**************************************************************************

      REAL FUNCTION F3(E)
C     COMPUTES X-INTEGRAND FOR CAVITY STREAMLINES
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      TS = 1.0
      IF(E.LT.0.0) TS = -1.0
      IF((E.GT.TA).OR.(E.LT.(-1.0))) GO TO  1
      S2 = 0.0
      IF(E.EQ.(-1.0))  R6 = 0.5*PI
      IF(E.EQ.TA)  R6 = 0.0
      GO TO 2
    1 R1 = SQRT((E-TA)/(1.0+E))
      R2 = SQRT((TE-TA)/(1.0+TE))
      R3 = 1.0/R1
      R4 = SQRT((1.0+TD)/(TD-TA))
      R5 = ALOG((R2+R1)/(TS*(R2-R1))) - ALOG((R3+R4)/(TS*(R3-R4)))
      S2 = 0.5*LNSIG*R5/PI
      R6 = ATAN(R1/STA)
    2 ZI = S2+AL-PI+2.0*R6
      IF(IG.GT.0) ZI = ZI + G2C(E) + G2S(E)
      R7 = COS(ZI)*E/(BETA-E)
      IF(IG.GT.0) R7 = R7/SQRT(1.0-YC(E)*FSFAC)
      F3 = R7
      RETURN
      END

**************************************************************************

      REAL FUNCTION F4(E)
C     COMPUTES Y-INTEGRAND FOR CAVITY STREAMLINES
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      TS = 1.0
      IF(E.LT.0.0) TS = -1.0
      IF((E.GT.TA).OR.(E.LT.(-1.0))) GO TO  1
      S2 = 0.0
      IF(E.EQ.(-1.0))  R6 = 0.5*PI
      IF(E.EQ.TA)  R6 = 0.0
```

```
      GO TO 2
    1 R1 = SQRT((E-TA)/(1.0+E))
      R2 = SQRT((TE-TA)/(1.0+TE))
      R3 = 1.0/R1
      R4 = SQRT((1.0+TD)/(TD-TA))
      R5 = ALOG((R2+R1)/(TS*(R2-R1))) - ALOG((R3+R4)/(TS*(R3-R4)))
      S2 = 0.5*LNSIG*R5/PI
      R6 = ATAN(R1/STA)
    2 ZI = S2+AL-PI+2.0*R6
      IF(IG.GT.0) ZI = ZI + G2C(E) + G2S(E)
      R7 = -SIN(ZI)*E/(BETA-E)
      IF(IG.GT.0) R7 = R7/SQRT(1.0-YC(E)*FSFAC)
      F4 = R7
      RETURN
      END
```

```
********************************************************************************
```

```
      REAL FUNCTION F5(E)
C     COMPUTES X-INTEGRAND ON WAKES AND FREE SURFACE
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      R1 = SQRT((E-TA)/(1.0+E))
      R2 = SQRT((TE-TA)/(1.0+TE))
      R3 = 1.0/R1
      R4 = SQRT((1.0+TD)/(TD-TA))
      R5 = ALOG((R1+R2)/(R1-R2)) - ALOG((R3+R4)/(R3-R4))
      R6 = ATAN(R1/STA)
      GAM = 0.5*R5*LNSIG/PI +AL-PI+2.0*R6
      IF(IG.GT.0) GAM = GAM + G3C(E) + G3S(E)
      R7 =  COS(GAM)*E/(BETA-E)
      IF(E.GT.BETA) R7 = R7/SQRT(1.0+2.0*(D-YS(E))/(FSQ*PL))
      F5 = R7
      RETURN
      END
```

```
********************************************************************************
```

```
      REAL FUNCTION F6(E)
C     COMPUTES Y-INTEGRAND ON WAKES AND FREE SURFACE
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      R1 = SQRT((E-TA)/(1.0+E))
      R2 = SQRT((TE-TA)/(1.0+TE))
      R3 = 1.0/R1
      R4 = SQRT((1.0+TD)/(TD-TA))
      R5 = ALOG((R1+R2)/(R1-R2)) - ALOG((R3+R4)/(R3-R4))
      R6 = ATAN(R1/STA)
      GAM = 0.5*R5*LNSIG/PI +AL-PI+2.0*R6
      IF(IG.GT.0) GAM = GAM + G3C(E) + G3S(E)
      R7 = -SIN(GAM)*E/(BETA-E)
      IF(E.GT.BETA) R7 = R7/SQRT(1.0+2.0*(D-YS(E))/(FSQ*PL))
      F6 = R7
      RETURN
```

```
      END

##########################################################################

      INTEGER FUNCTION  NEWCO(A,B,FUNCT,RESULT,TOLER,STEP)
C*************************************************************************
C*************************************************************************
C*                                                                       *
C*         THIS FUNCTION SUBPROGRAM INTEGRATES A FUNCTION BETWEEN THE    *
C*    BETWEEN GIVEN LIMITS USING THE NEWTON-COTES FIVE POINT FORMULA     *
C*    (SEE HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-      *
C*    HILL, P71 AND FF).  THE INTEGRATION IS PERFORMED BY BREAKING       *
C*    THE INTERVAL INTO SUCCESSIVELY SMALLER SUBINTERVALS AND            *
C*    INTEGRATING EACH USING THE REFERENCE FORMULA.  THE SUBPROGRAM      *
C*    RETURNS WHEN THE PERCENT DIFFERENCE BETWEEN THE RESULTS FROM TWO   *
C*    ADJACENT ITERATIONS  IS LESS THAN A STATED TOLERENCE OR THE        *
C*    INTERVAL WIDTH IS LESS THAN A GIVEN MINIMUM.                       *
C*         THE CALL IS                                                   *
C*              I = NEWCO(A,B,FUNCT,RESULT,TOLER,STEP)  WHERE            *
C*                   A  IS  THE LOWER BOUND OF INTEGRATION               *
C*                   B  IS  THE UPPER BOUND OF INTEGRATION               *
C*               FUNCT  IS  THE FUNCTION TO BE INTEGRATED                *
C*              RESULT  IS  THE VALUE OF THE INTEGRAL                    *
C*                 TOLER IS THE ACCURACY DESIRED (SEE ABOVE)             *
C*                 STEP  IS THE MINIMUM ALLOWABLE STEP (SEE ABOVE)       *
C*                                                                       *
C*         I IS SET EQUAL TO                                             *
C*                1 IF THE RESULT  RETURNED WITHIN THE TOLERENCE         *
C*                2 IF THE MINIMUM STEP WAS EXCEEDED                     *
C*                                                                       *
C*         IN BOTH CASES RESULT CONTAIN THE VALUE OF THE INTEGRAL TO     *
C*         THAT POINT.                                                   *
C*                                                                       *
C*************************************************************************
C*************************************************************************
      AMB   = B - A
      IF(AMB.EQ.0.0) GO TO 101
C*************************************************************************
C*    INITIALIZE ITERATION                                               *
C*************************************************************************
      ODDOLD = FUNCT((A+B)/2.0)
      EVENS =(FUNCT(A)+FUNCT(B))/2.0 + ODDOLD
      NN =2
  1   H2=  AMB/FLOAT(   NN)
      X = A + H2/2.0
      SUM = 0.0
C*************************************************************************
C*    ADD UP ODD TERMS                                                   *
C*************************************************************************
      DO 2 I = 1,NN
      SUM = SUM + FUNCT(X)
  2   X= X+ H2
      ODDNEW = 2.0*SUM/FLOAT(NN)
C*************************************************************************
C*    COMPUTE RESULT = (7*EVEN TERMS - OLD ODD TERMS + 16*NEW ODD        *
C*          TERMS)/45 =                                                  *
C*    SUM (2H(7F0+32F1+12F2+32F3+7F4)/45 OVER ALL INTERVALS              *
C*************************************************************************
```

```
      ANSWER =  AMB*(7.0*EVENS - ODDOLD + 16.0* ODDNEW)/45.0
      IF(ANSWER.EQ.0.0) GO TO  101
      IF(NN.EQ.2) GO TO  3
C***********************************************************************
C*    COMPARE CURRENT RESULT WITH OLD RESULT AND RETURN IF WITHIN      *
C*         TOLERANCE ELSE DIVIDE INTO NEW INTERVALS AND ITERATE        *
C***********************************************************************
      IF(ABS((ANSWER-RESULT)/RESULT).LE.TOLER)  GO TO  100
  3   RESULT = ANSWER
      NN = NN + 2
       EVENS = (EVENS+ ODDNEW)/2.0
      ODDOLD = ODDNEW/2.0
      IF(ABS(H2).GT.STEP)  GO TO  1
      NEWCO  =  2
      RETURN
101   RESULT = 0.0
100   NEWCO  = 1
      RETURN
      END
```

```
*************************************************************************
```

```
      FUNCTION ARGZ(X,Y)
C     COMPUTES ARGUMENT OF Z=X+IY, -PI.LT.ARG.LE.PI
      IF(X) 1,2,3
    3 ARGZ = ATAN(Y/X)
      RETURN
    2 IF(Y) 4,5,6
    4 ARGZ = - 1.57079633
      RETURN
    5 ARGZ = 0.0
      RETURN
    6 ARGZ = 1.57079633
      RETURN
    1 IF(Y.GE.0.0) ARGZ = 3.141592654 + ATAN(Y/X)
      IF(Y.LT.0.0) ARGZ =-3.141592654 + ATAN(Y/X)
      RETURN
      END
```

```
*************************************************************************:
```

```
      REAL FUNCTION XARC(X)
C     COMPUTES X-ARC INTEGRAND
      COMMON TA,TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      Z=X
      CALL ARC(Z, ZR, ZI)
      A = (BETA+DEL*COS(Z))*SIN(ZI) - DEL*SIN(Z)*COS(ZI)
      XARC = A/EXP(ZR)
      RETURN
      END
```

```
*************************************************************************.
```

```
      REAL FUNCTION YARC(X)
C     COMPUTES Y-ARC INTEGRAND
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      Z=X
      CALL ARC(Z, ZR, ZI)
      A = (BETA+DEL*COS(Z))*COS(ZI) + DEL*SIN(Z)*SIN(ZI)
      YARC = A/EXP(ZR)
      RETURN
      END

********************************************************************************

      SUBROUTINE ARC(X, ZR, ZI)
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      DC = DEL*COS(X)
      DS = DEL*SIN(X)
      XA = BETA-TA+DC
      XB = BETA+1.0+DC
      YA = DS
      XDNM = XB*XB + YA*YA
      XC = (XA*XB+YA*YA)/XDNM
      YC = YA*(XB-XA)/XDNM
      SC = 0.5*SQRT(XC*XC + YC*YC)
      XD = SQRT( 0.5*XC+ SC)
      YD = SQRT(-0.5*XC+ SC)
      IF(YC.LT.0.0) XD = -XD
      XDNM = XA*XA+YA*YA
      XE = (XA*XB+YA*YA)/XDNM
      YE = YA*(XA-XB)/XDNM
      SE = 0.5*SQRT(XE*XE+YE*YE)
      XF = SQRT( 0.5*XE + SE)
      YF = SQRT(-0.5*XE + SE)
      IF(YE.LT.0.0) XF = -XF
      XG = XD/STA
      YG = YD/STA
      TI = 0.5*(ALOG(XG*XG+(1.0+YG)**2) - ALOG(XG*XG+(1.0-YG)**2))
      TR = - ARGZ(XG,(1.0+YG)) + ARGZ(-XG,(1.0-YG))
      XH = XB*XA - YA*YA
      YH = YA*(XA+XB)
      SH = 0.5*SQRT(XH*XH + YH*YH)
      XJ = SQRT( 0.5*XH + SH)
      YJ = SQRT(-0.5*XH + SH)
      IF(YH.LT.0.0) XJ = -XJ
C     DETERMINE G3C HERE
      G3CR = 0.0
      IF(IG.GT.0) G3CR = GIRI(X)
      G3CI = 0.0
      IF(IG.GT.0) G3CI = GIII(X)
C
C     DETERMINE G3S HERE
      G3SR = 0.0
      IF(IG.GT.0) G3SR = GKRI(X)
      G3SI = 0.0
```

```
      IF(IG.GT.0) G3SI = GKII(X)
C
      X1 = XD + R2
      X2 = XD - R2
      X3 = XF + R4
      X4 = XF - R4
      SR = ALOG(X1*X1+YD*YD) - ALOG(X2*X2+YD*YD) - ALOG(X3*X3+YF*YF) +
     1ALOG(X4*X4+YF*YF)
      SR = 0.25*LNSIG*SR/PI
      SI = 0.5*LNSIG*(ARGZ(X1,YD) - ARGZ(X2,YD) - ARGZ(X3,YF) + ARGZ(X4,
     1YF))/PI
      ZR = -SI -TI -G3CI -G3SI
      ZI = SR +AL -PI +TR +G3CR +G3SR
      RETURN
      END
```

##:

```
      REAL FUNCTION YC(X)
      COMMON TA TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C2/ TLC, TUC, YLC, YUC
      DIMENSION TLC(100),TUC(100),YLC(100),YUC(100)
      COMMON /C3/ ENLC,ENUC,ENFS,NLC,NUC,NFS
      IF(X) 1,1,2
C     GIVES  YC(X)  FOR X.LE.-1.0
    1 J = 1.0 + (1.0+X)*FNLC/(1.0+TD)
      IF(J.GE.NLC)  J=NLC-1
      YC = YLC(J) + (X-TLC(J))*(YLC(J+1)-YLC(J))/(TLC(J+1)-TLC(J))
      RETURN
C     GIVES YC(X) FOR X.GE.TA
    2 IF(X.GT.(3.0*TA))  GO TO 3
      J = 1.0 + 2.5*(X/TA - 1.0)
      GO TO 4
    3 J = 6.0 + (X-3.0*TA)*(ENUC-5.0)/(TF-3.0*TA)
      IF(J.GE.NUC)  J=NUC-1
    4 YC = YUC(J) + (X-TUC(J))*(YUC(J+1)-YUC(J))/(TUC(J+1)-TUC(J))
      RETURN
      END
```

##:

```
      REAL FUNCTION YS(X)
      COMMON TA,TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      COMMON /C3/ ENLC,FNUC,FNFS,NLC,NUC,NFS
      COMMON /C4/ TFS, YFS
      DIMENSION TFS(100),YFS(100)
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      IF(X.GT.TFS(5)) GO TO 2
      J = 10.0*(X-BETA)/(BETA-TE)
      IF(J.LT.1)  GO TO 4
      GO TO 3
    2 J = 5.0 + (FNFS-4.0)*     ((X-TFS(5))/(TU-BETA-DEL))**0.333333
      IF(J.GE.NFS)  J=NFS-1
```

```
    3 YS = YFS(J) + (X-TFS(J))*(YFS(J+1)-YFS(J))/(TFS(J+1)-TFS(J))
      RETURN
    4 YS = D + (X-BETA   )*(YFS(1)-D)/(TFS(1)-BETA)
      RETURN
      END

************************************************************************

      REAL FUNCTION F7(X)
C     INTEGRAND FOR GC(T)
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0-YC(X)*FSFAC)
      B = (X-T)*SQRT((1.0+X)*(X-TA))
      F7 = A/B
      RETURN
      END

************************************************************************

      REAL FUNCTION F8(X)
C     INTEGRAND FOR GS(T)
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0+2.0*(D-YS(X))/(FSQ*PL))
      B = (X-T)*SQRT((1.0+X)*(X-TA))
      F8 = A/B
      RETURN
      END

************************************************************************

      REAL FUNCTION G1C(X)
      COMMON /C6/ TCF, GCF, GSF
      DIMENSION TCF(11),GCF(11),GSF(11)
      J = 1.0 + 10.0*(X+1.0)/(TCF(11)+1.0)
      IF(J.GT.10) J=10
      G1C = GCF(J) + (GCF(J+1)-GCF(J))*(X-TCF(J))/(TCF(J+1)-TCF(J))
      RETURN
      END

************************************************************************

      REAL FUNCTION G1S(X)
      COMMON /C6/ TCF, GCF, GSF
      DIMENSION TCF(11),GCF(11),GSF(11)
      J = 1.0 + 10.0*(X+1.0)/(TCF(11)+1.0)
      IF(J.GT.10) J=10
      G1S = GSF(J) + (GSF(J+1)-GSF(J))*(X-TCF(J))/(TCF(J+1)-TCF(J))
      RETURN
```

```
      END

******************************************************************************

      REAL FUNCTION G2C(X)
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C7/ G2CU,G2SU,G2CL,G2SL
      DIMENSION G2CU(100),G2SU(100),G2CL(100),G2SL(100)
      COMMON /C2/ TLC, TUC, YLC, YUC
      DIMENSION TLC(100),TUC(100),YLC(100),YUC(100)
      COMMON /C3/ ENLC,ENUC,ENFS,NLC,NUC,NFS
      IF(X.LT.0.0) GO TO 2
C     UPPER CAVITY BOUNDARY
      IF(X.GT.(3.0*TA)) GO TO 3
      J = 1.0 + 2.5*(X/TA - 1.0)
      GO TO 4
    3 J = 6.0 + (X-3.0*TA)*(ENUC-5.0)/(TE-3.0*TA)
      IF(J.GE.NUC) J=NUC-1
    4 G2C = G2CU(J)+(X-TUC(J))*(G2CU(J+1)-G2CU(J))/(TUC(J+1)-TUC(J))
      RETURN
C     LOWER CAVITY BOUNDARY
    2 J = 1.0 + (1.0+X)*ENLC/(1.0+TD)
      IF(J.GE.NLC) J=NLC-1
      G2C = G2CL(J)+(X-TLC(J))*(G2CL(J+1)-G2CL(J))/(TLC(J+1)-TLC(J))
      RETURN
      END

******************************************************************************

      REAL FUNCTION G2S(X)
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C7/ G2CU,G2SU,G2CL,G2SL
      DIMENSION G2CU(100),G2SU(100),G2CL(100),G2SL(100)
      COMMON /C2/ TLC, TUC, YLC, YUC
      DIMENSION TLC(100),TUC(100),YLC(100),YUC(100)
      COMMON /C3/ ENLC,ENUC,ENFS,NLC,NUC,NFS
      IF(X.LT.0.0) GO TO 2
C     UPPER CAVITY BOUNDARY
      IF(X.GT.(3.0*TA)) GO TO 3
      J = 1.0 + 2.5*(X/TA - 1.0)
      GO TO 4
    3 J = 6.0 + (X-3.0*TA)*(ENUC-5.0)/(TE-3.0*TA)
      IF(J.GE.NUC) J=NUC-1
    4 G2S = G2SU(J)+(X-TUC(J))*(G2SU(J+1)-G2SU(J))/(TUC(J+1)-TUC(J))
      RETURN
C     LOWER CAVITY BOUNDARY
    2 J = 1.0 + (1.0+X)*ENLC/(1.0+TD)
      IF(J.GE.NLC) J=NLC-1
      G2S = G2SL(J)+(X-TLC(J))*(G2SL(J+1)-G2SL(J))/(TLC(J+1)-TLC(J))
      RETURN
      END

******************************************************************************
```

```
      REAL FUNCTION G3C(X)
      COMMON TA,TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C3/ ENLC,FNUC,FNFS,NLC,NUC,NFS
      COMMON /C4/ TFS, YFS
      DIMENSION TFS(100),YFS(100)
      COMMON /C8/ WL,WU,GCWU,GSWU,GCWL,GSWL
      DIMENSION WL(20),WU(10),GCWU(10),GSWU(10),GCWL(20),GSWL(20)
      COMMON /C9/ JWLM
      COMMON /C10/ GCFS, GSFS
      DIMENSION GCFS(100), GSFS(100)
      IF(X.LT.0.0)  GO TO  2
      IF(X.GT.BETA)  GO TO 3
C     UPPER WAKE
      J = 1.0 + 10.0*(X-TE)/(BETA-TE)
      IF(J.GT.9)  J=9
      G3C = GCWU(J) + (X-WU(J))*(GCWU(J+1)-GCWU(J))/(WU(J+1)-WU(J))
      RETURN
C     LOWER WAKE
    2 J = 1
      IF(X.GT.WL(JWLM)) GO TO 5
      J = JWLM - 1
      GO TO  6
    5 IF((X.LE.WL(J)).AND.(X.GT.WL(J+1))) GO TO 4
      J = J + 1
      GO TO  5
    4 IF(J.GE.JWLM) J = JWLM - 1
    6 G3C = GCWL(J) + (X-WL(J))*(GCWL(J+1)-GCWL(J))/(WL(J+1)-WL(J))
      RETURN
C     FREE SURFACE
    3 IF(X.LT.TFS(5)) GO TO 7
      J=5.0 + (ENFS-4.0)*((X-TFS(5))/(TFS(NFS)-TFS(5)))**0.33333333
      IF(J.GE.NFS) J=NFS-1
      GO TO 8
    7 J = 10.0*(X-BETA)/(BETA-TE)
    8 G3C = GCFS(J) + (GCFS(J+1)-GCFS(J))*(X-TFS(J))/(TFS(J+1)-TFS(J))
      RETURN
      END
```

****

```
      REAL FUNCTION G3S(X)
      COMMON TA,TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C3/ ENLC,ENUC,ENFS,NLC,NUC,NFS
      COMMON /C4/ TFS, YFS
      DIMENSION TFS(100),YFS(100)
      COMMON /C8/ WL,WU,GCWU,GSWU,GCWL,GSWL
      DIMENSION WL(20),WU(10),GCWU(10),GSWU(10),GCWL(20),GSWL(20)
      COMMON /C9/ JWLM
      COMMON /C10/ GCFS, GSFS
      DIMENSION GCFS(100), GSFS(100)
      IF(X.LT.0.0)  GO TO  2
      IF(X.GT.BETA)  GO TO 3
C     UPPER WAKE
      J = 1.0 + 10.0*(X-TE)/(BETA-TE)
```

```
      IF(J.GT.9)  J=9
      G3S = GSWU(J) + (X-WU(J))*(GSWU(J+1)-GSWU(J))/(WU(J+1)-WU(J))
      RETURN
C     LOWER WAKE
    2 J = 1
      IF(X.GT.WL(JWLM)) GO TO 5
      J = JWLM - 1
      GO TO  6
    5 IF((X.LE.WL(J)).AND.(X.GT.WL(J+1))) GO TO 4
      J = J + 1
      GO TO  5
    4 IF(J.GE.JWLM) J = JWLM - 1
    6 G3S = GSWL(J) + (X-WL(J))*(GSWL(J+1)-GSWL(J))/(WL(J+1)-WL(J))
      RETURN
C     FREE SURFACE
    3 IF(X.LT.TFS(5)) GO TO 7
      J=5.0 + (ENFS-4.0)*((X-TFS(5))/(TFS(NFS)-TFS(5)))**0.33333333
      IF(J.GE.NFS) J=NFS-1
      GO TO 8
    7 J = 10.0*(X-BETA)/(BETA-TE)
    8 G3S = GSFS(J) + (GSFS(J+1)-GSFS(J))*(X-TFS(J))/(TFS(J+1)-TFS(J))
      RETURN
      END
```

\#

```
      REAL FUNCTION F9(X)
      COMMON TA,TO,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0-YC(X)*FSFAC)
      B = X-BETA-DEL*COS(T)
      C = DEL*SIN(T)
      E = B/(B*B+C*C)
      B = SQRT((1.0+X)*(X-TA))
      F9  =  A*E/B
      RETURN
      END
```

\#

```
      REAL FUNCTION F10(X)
      COMMON TA,TO,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0-YC(X)*FSFAC)
      B = X-BETA-DEL*COS(T)
      C = DEL*SIN(T)
      E = C/(B*B+C*C)
      B = SQRT((1.0+X)*(X-TA))
      F10 =  A*E/B
      RETURN
      END
```

```
************************************************************************

      REAL FUNCTION F11(X)
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0+2.0*(D-YS(X))/(FSQ*PL))
      B = X-BETA-DEL*COS(T)
      C = DEL*SIN(T)
      E = B/(B*B+C*C)
      B = SQRT((1.0+X)*(X-TA))
      F11 =  A*E/B
      RETURN
      END

************************************************************************

      REAL FUNCTION F12(X)
      COMMON TA,TD,TE,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C1/ DEL, R2, R4
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0+2.0*(D-YS(X))/(FSQ*PL))
      B = X-BETA-DEL*COS(T)
      C = DEL*SIN(T)
      E = C/(B*B+C*C)
      B = SQRT((1.0+X)*(X-TA))
      F12 =  A*E/B
      RETURN
      END

************************************************************************

      REAL FUNCTION GIRI(X)
      COMMON /C11/ GIR,GII,GKR,GKI,THE
      DIMENSION GIR(6),GII(6),GKR(6),GKI(6),THE(6)
      J = 1.0 + 5.0*X/3.141592654
      IF(J.GT.5) J=5
      GIRI = GIR(J) + (GIR(J+1)-GIR(J))*(X-THE(J))/0.6283185308
      RETURN
      END

************************************************************************

      REAL FUNCTION GIII(X)
      COMMON /C11/ GIR,GII,GKR,GKI,THE
      DIMENSION GIR(6),GII(6),GKR(6),GKI(6),THE(6)
      J = 1.0 + 5.0*X/3.141592654
      IF(J.GT.5) J=5
      GIII = GII(J) + (GII(J+1)-GII(J))*(X-THE(J))/0.6283185308
      RETURN
```

```
      END

**************************************************************************************

      REAL FUNCTION GKRI(X)
      COMMON /C11/ GIR,GII,GKR,GKI,THE
      DIMENSION GIR(6),GII(6),GKR(6),GKI(6),THE(6)
      J = 1.0 + 5.0*X/3.141592654
      IF(J.GT.5) J=5
      GKRI = GKR(J) + (GKR(J+1)-GKR(J))*(X-THE(J))/0.6283185308
      RETURN
      END

**************************************************************************************

      REAL FUNCTION GKII(X)
      COMMON /C11/ GIR,GII,GKR,GKI,THE
      DIMENSION GIR(6),GII(6),GKR(6),GKI(6),THE(6)
      J = 1.0 + 5.0*X/3.141592654
      IF(J.GT.5) J=5
      GKII = GKI(J) + (GKI(J+1)-GKI(J))*(X-THE(J))/0.6283185308
      RETURN
      END

**************************************************************************************

      REAL FUNCTION F13(X)
C     USED IN COMPUTING B2G
      COMMON TA,TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0-YC(X)*FSFAC)
      B = SQRT((1.0+X)*(X-TA))
      F13 = A/B
      RETURN
      END

**************************************************************************************

      REAL FUNCTION F14(X)
C     USED IN COMPUTING B2G
      COMMON TA,TD,TF,LNSIG,BETA,PI,AL,STA,TU,IG
      REAL LNSIG
      COMMON /C5/ T, FSFAC,D,FSQ,PL
      A = ALOG(1.0+2.0*(D-YS(X))/(FSQ*PL))
      B = SQRT((1.0+X)*(X-TA))
      F14 = A/B
      RETURN
      END

**************************************************************************************
```

```
NUMBER OF ERRORS DETECTED = 0000.  NUMBER OF CARDS = 01577.
COMPILATION TIME = 00061 SECONDS ELAPSED.  00028.16 SECONDS PROCESSIN
D2 STACK SIZE = 00061 WORDS.  FILESIZE = 00140 WORDS.
TOTAL PROGRAM CODE = 05101 WORDS.  ARRAY STORAGE = 01941 WORDS.
NUMBER OF PROGRAM SEGMENTS = 0043.  NUMBER OF DISK SEGMENTS = 00001.
ESTIMATED CORE STORAGE REQUIREMENT = ~~000000~~ WORDS.
                                     010000
```

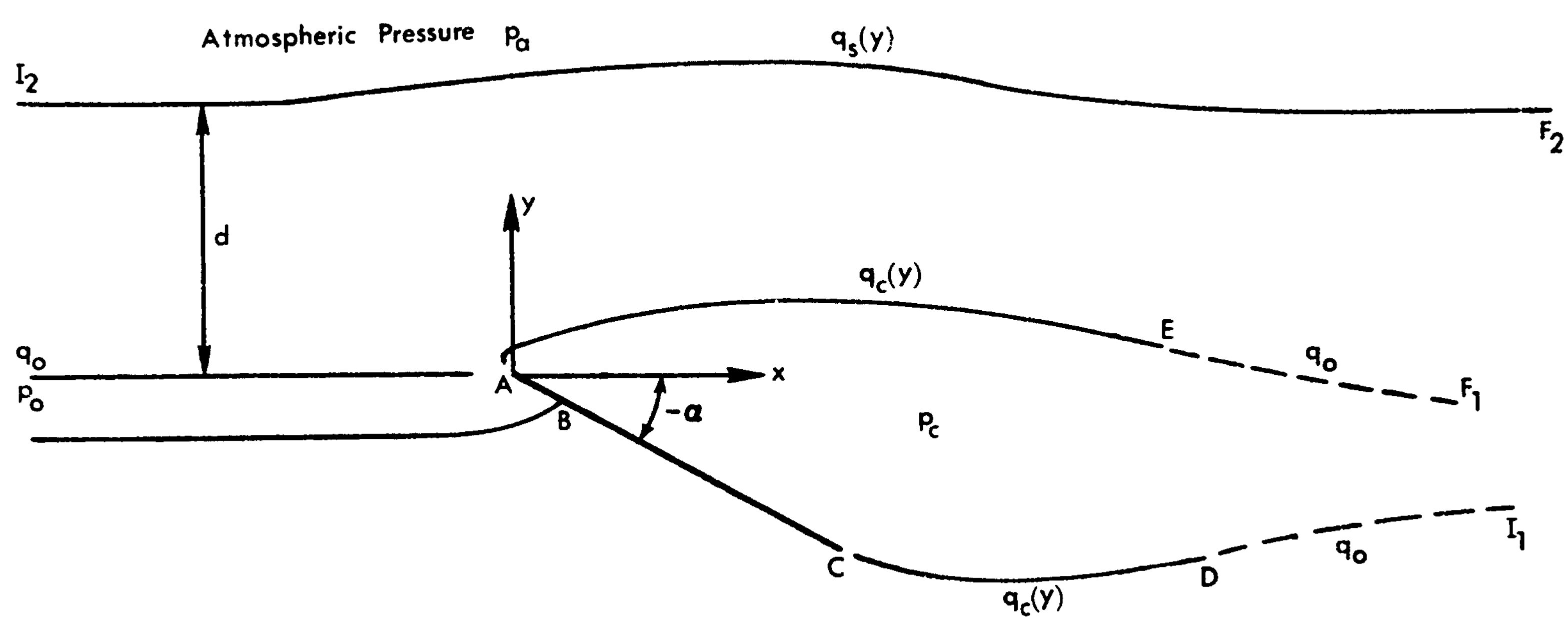

Figure 1. The Physical Plane, Double-spiral-vortex Model

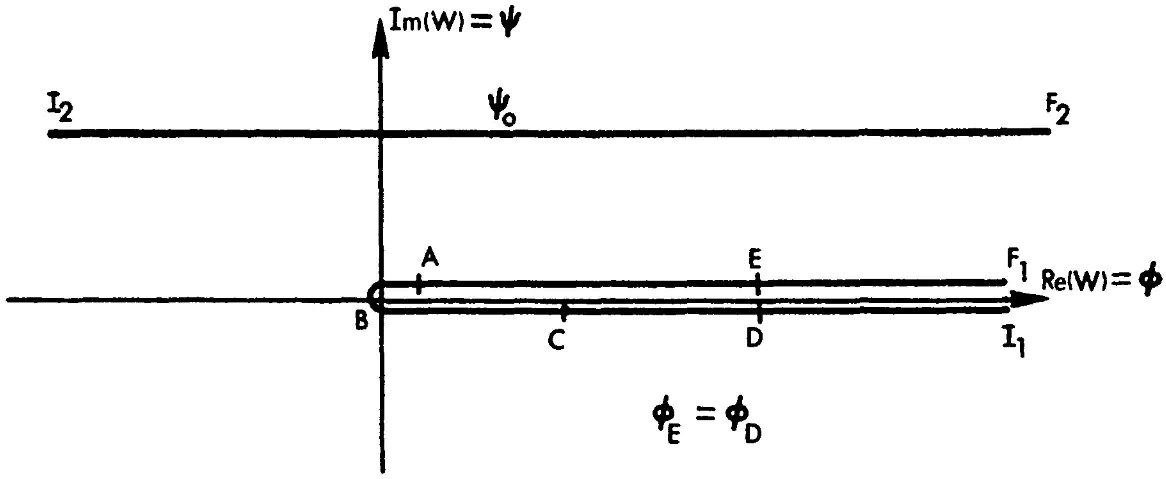

Figure 2. The Complex Potential Plane

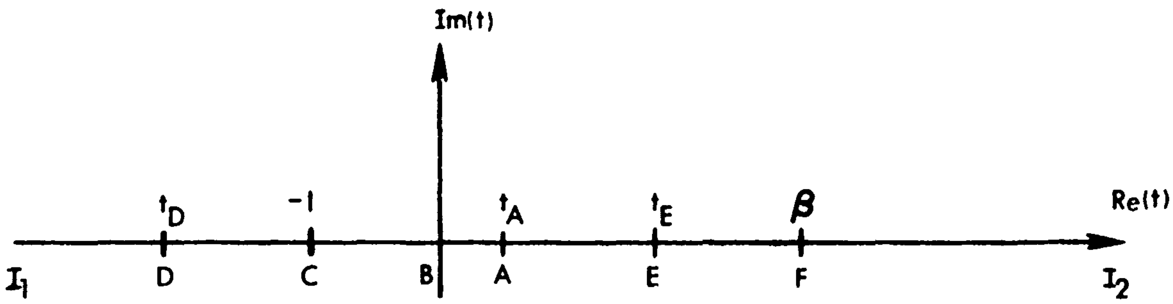

Figure 3. The t-plane